BEST AVAILABLE IMAGES

Defective images within this document are accurate representations of the original documents submitted by the applicant.

Defects in the images may include (but are not limited to):

- BLACK BORDERS
- TEXT CUT OFF AT TOP, BOTTOM OR SIDES
- FADED TEXT
- ILLEGIBLE TEXT
- SKEWED/SLANTED IMAGES
- COLORED PHOTOS
- BLACK OR VERY BLACK AND WHITE DARK PHOTOS
- GRAY SCALE DOCUMENTS

IMAGES ARE BEST AVAILABLE COPY.

As rescanning documents *will not* correct images, please do not report the images to the Image Problem Mailbox.

(19) Europäisches Patentamt
European Patent Office
Office européen des brevets

(11) Publication number : 0 631 244 A2

(12) EUROPEAN PATENT APPLICATION

(21) Application number : 94302927.2

(51) Int. Cl.[5] : G06F 15/403

(22) Date of filing : 25.04.94

(30) Priority : 24.06.93 US 82938

(43) Date of publication of application :
28.12.94 Bulletin 94/52

(84) Designated Contracting States :
DE FR GB

(71) Applicant : XEROX CORPORATION
Xerox Square
Rochester New York 14644 (US)

(72) Inventor : Kuplec, Julian M.
10070 Craft Drive
Cupertino, California 95014 (US)

(74) Representative : Johnson, Reginald George et al
Rank Xerox Ltd
Patent Department
Parkway
Marlow Buckinghamshire SL7 1YL (GB)

(54) A method and system of information retrieval.

(57) A computerized method for organizing information retrieval based on the content of a set of primary documents. The method generates answer hypotheses based on text found in the primary documents and, typically, a natural-language input string such as a question. The answer hypotheses can include phrases or words not present in the input string. Answer hypotheses are verified and ranked based on their verification evidence. A text corpus (12) can be queried to provide verification evidence not present in the primary documents. In another aspect the method is implemented in the context of a larger two-phase method, of which the first phase comprises the method of the invention and the second phase of the method comprises answer extraction.

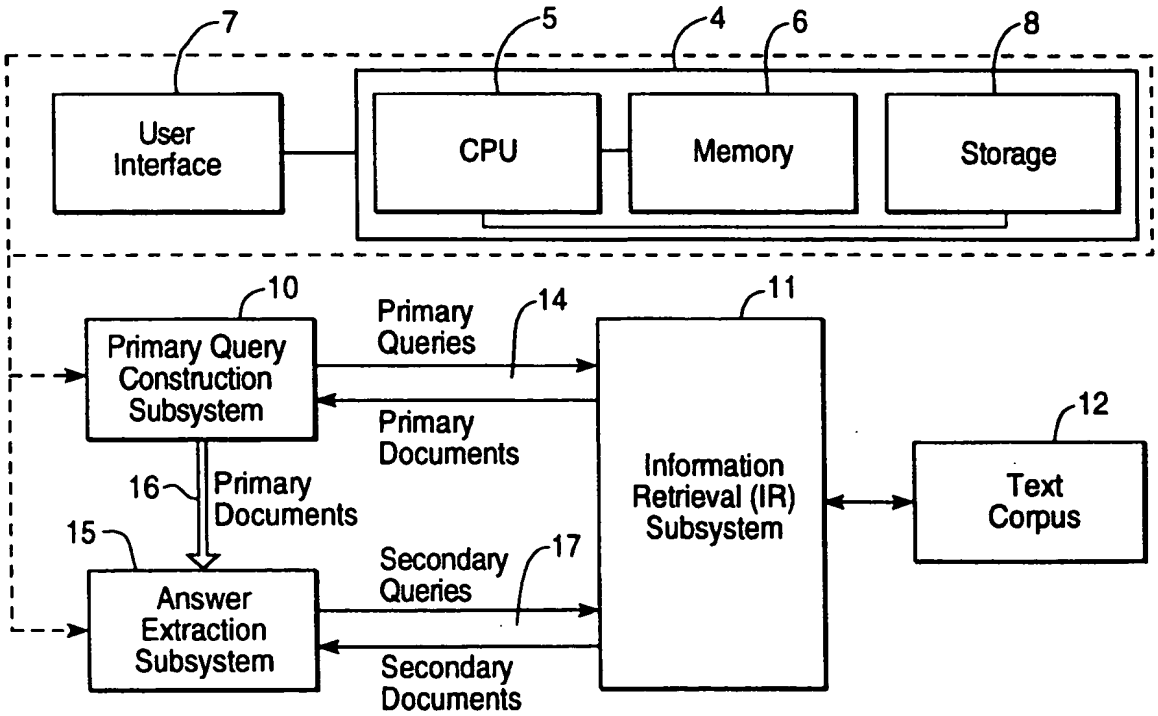

FIG. 1

EP 0 631 244 A2

The present invention relates to computerized information retrieval (IR) systems and methods, and more particularly to computerized information-retrieval systems and methods used with textual databases.

In prior art IR systems and methods the user often has to proceed iteratively, usually using a special query language, by beginning with an initial query, scanning the results of that query (that is, the documents that were retrieved in response to that query), modifying the query if the results were unsatisfactory, and repeating these steps until satisfactory results are obtained. The user is responsible for modifying or refining the query and typically receives little or no help from the system in query construction or refinement.

Prior art systems cannot locate or highlight phrases within documents if the phrases are not found in the input query. For example, in response to a user query regarding what film pits Humphrey Bogart against gangsters in the Florida Keys, prior art IR systems would accept from the user a query containing words such as "film," "Bogart," "gangster," and "Florida Keys," and would search for the co-occurrence of these words within single documents. It would be left to the user to piece together the various documents thus retrieved to determine the correct answer.

Attempts have been made in the prior art to allow users to formulate queries in natural language. One such attempt is found in systems called question-answering systems. A question-answering system can respond to certain natural-language queries, but only because the data in its database are heavily preprocessed to accommodate such queries. For example, a question-answering system designed to respond to user queries about moon rocks will typically employ a database configured as a two-dimensional array, with moon-rock sample numbers arrayed along one dimension and fact categories, such as ore content of the rocks, arrayed along the other axis. The system responds to user queries simply by accessing the appropriate cell of the array.

Prior art IR systems and methods do not automatically perform a sequence of queries designed to include the optimal query or queries needed to answer a user's natural-language question. Furthermore, prior art IR systems and methods neither guess at the answer to the user's question nor attempt to manipulate or process queries based on such a guess. They simply match text. There is therefore a need to provide an improved IR system and method which does not have the disadvantages found in the prior art. The present invention strives to meet this need. Accordingly, the present invention provides a method for information retrieval according to any of the appended claims.

The present invention provides a method for organizing information retrieval based on the content of a set of documents. In one embodiment the method generates answer hypotheses based on text found in the documents of the set and, typically, a natural-language input string such as a question. The answer hypotheses can include phrases or words not present in the input string. In some embodiments, such as those described up to this point, the answer hypotheses are verified and can be ranked based on their verification evidence. If verification is performed, a text corpus can be used to provide verification evidence not present in the set of documents. The documents can be present in the text corpus, but need not be. In some embodiments certain steps of the method, such as verification, can be omitted. In some embodiments the input string can be omitted.

In another embodiment the present invention is used in the context of a larger, two-phase method for information retrieval in response to a user-supplied natural-language input string such as a question. The first phase comprises primary query construction, in which a set of documents likely to be relevant is retrieved from a text corpus. The second phase comprises the method of the present invention, which is used to analyze documents retrieved in the first phase to generate answer hypotheses that are likely to be the answer to the question and to select and point out to the user the document or documents most likely to contain the answer to the user's question.

The present invention will be described further, by way of examples, with reference to the accompanying drawings, in which:-

Fig. 1 illustrates a system suitable for an embodiment of the present invention;

Fig. 2 is a high-level flow chart diagramming the method of the present invention;

Fig. 3 flowcharts the component steps of question processing in one embodiment of the method;

Fig. 4 flowcharts the component steps of preliminary hypothesis generation in one embodiment of the method;

Fig. 5 flowcharts the steps of hypothesis verification in one embodiment of the method; and

Fig. 6 flowcharts the component steps of hypothesis ranking in one embodiment of the method.

The disclosures in this application of all articles and references, including patent documents, are incorporated herein by reference.

The description that follows is in four parts. Part I is an introduction that includes an overview of the invention, an example of how it works, and a glossary of terms. Part II describes the method of the present invention in one embodiment. Part III describes MURAX, a system designed to embody the method of the present invention in the context of a larger two-phase method of which the method of the present invention is used to carry out the second phase. Part IV concludes the description.

PART I. INTRODUCTION

1. Overview of the Invention

The present invention provides a method for answer extraction. A system operating according to this method accepts a natural-language input string such as a user-supplied question and a set of relevant documents that are assumed to contain the answer to the question. In response, it generates answer hypotheses and finds these hypotheses within the documents. In the Humphrey Bogart example above, the method of the present invention would generate the answer hypothesis "Key Largo" and find that hypothesized term in one or more documents previously retrieved from an IR system or subsystem.

The present invention can, in the context of a question that asks for a specific answer, provide an answer that is nowhere to be found in the question. This greatly facilitates the user's finding the desired answer. The user does not have to piece together the answer by hunting through the documents. Instead, the invention highlights the answer hypothesis, in this case "Key Largo," in the retrieved documents for the user's convenience. The invention can draw on and combine information from multiple retrieved documents in order to answer the user's question.

In the present invention, the answer hypotheses generated are used in new queries that the system constructs without user aid. These secondary queries are based on guesses--words that in the vast majority of cases are not present in the user's original question.

The method of the present invention can be used by itself or in a larger context. In particular, it can be used in a two-phase method that accepts a user's question in natural-language form and responds by not only producing relevant documents, but by also generating answer hypotheses and finding these hypotheses within the documents. Such a two-phase method comprises a first phase of primary query construction to retrieve documents likely to contain the answer to the question, and a second phase of answer extraction according to the method of the present invention.

2. Illustrative Example

It is helpful to present an illustrative example of answer extraction as performed according to the method of the invention in one embodiment. In this example, which will be called Example 1, the user's question is, "What Pulitzer Prize-winning novelist ran for mayor of New York City?"

The method begins by accepting as input the user's question and a set of documents that are assumed to contain the answer to the question. Typically the documents are retrieved from a database in response to IR queries that are carried out ahead of time. The documents of the set are called the primary documents.

The method then analyzes the question to detect the noun phrases that it contains. In this example, the noun phrases are "Pulitzer Prize," "novelist," "mayor," and "New York City." The method assumes that the documents contain some or all these noun phrases. This will be the case if the IR queries used to retrieve the primary documents are constructed based on the noun phrases.

Once the noun phrases are determined, the method looks at the sentences in which these noun phrases appear in the retrieved documents. These sentences are analyzed to detect additional noun phrases that appear in conjunction with the noun phrases of the original question. The method treats the additional noun phrases as preliminary answer hypotheses, and assumes that one of these preliminary hypotheses will eventually turn out to be the best answer to the user's question. Continuing with Example 1, suppose that the retrieved documents contain the following additional noun phrases in proximity to the noun phrase "New York City":

"Edith Wharton" (who was born in New York City)
"William Faulkner" (who visited New York City)
"Norman Mailer" (who lived in New York City)

Then "Edith Wharton," "William Faulkner," and "Norman Mailer" become the preliminary hypotheses for Example 1.

The preliminary hypotheses are scored and then ranked according to their scores. The scoring and ranking can be done using a simple criterion such as how many noun phrase words from the question appear within each document, with extra weight being given to the most important nouns (head nouns) in each noun phrase. Suppose that the documents containing the names "Edith Wharton" and "William Faulkner" refer to these persons as "novelists," but that the document containing the name "Norman Mailer" refers to Mr. Mailer as a "writer." Suppose also that all the retrieved documents include the noun phrases "New York City" and "Pulitzer Prize" and that the Mailer document includes the noun phrase "mayor." According to the simple scoring criterion, the documents are scored and ranked equally because the Wharton and Faulkner documents contain three of the noun phrases of the original question ("New York City," "Pulitzer Prize," and "novelist") as does the Mailer

document("New York City," "Pulitzer Prize," and "mayor").

After scoring and ranking the preliminary hypotheses, the method selects the top-ranked preliminary hypotheses and attempts to verify them. For each selected hypothesis, the method attempts to gather additional evidence to support the conclusion that the hypothesis is the correct answer. In effect the method asks, What does this hypothesis have in common with the original question? Verification is accomplished by lexicosyntactic analysis which looks for certain patterns in the user's question and attempts to find corresponding or related patterns in documents. If the corresponding or related patterns are found in conjunction with a hypothesis, this is considered evidence in favor of the hypothesis.

Often, verification cannot be accomplished using the primary documents alone. When the primary documents do not provide sufficient evidence to verify a hypothesis, the method attempts to find such evidence elsewhere in the database. Specifically, the method constructs additional IR queries called secondary queries and executes the secondary queries to retrieve additional documents called secondary documents. Several secondary queries can be run for each hypothesis, as needed.

Suppose that in Example 1 all three hypotheses are selected for verification. The primary documents indicate that Norman Mailer was from New York City, won a Pulitzer Prize, and ran for mayor, but do not indicate that he was a novelist. However, a secondary queries that seeks "Norman Mailer" in conjunction with "novelist" successfully retrieve secondary documents. Lexico-syntactic analysis of these documents verifies that Norman Mailer is a "novelist" as well as a "writer."

After all verification attempts are complete, the method rescores the hypotheses according to the degree to which they were successfully verified. In Example 1, Norman Mailer emerges as the winning answer hypothesis. Finally, the winning answer hypothesis can be presented to the user in conjunction with the documents and sentences in which it was found and the noun phrases that were used to verify it. In this way, the method shows not only what the answer is but why it was chosen.

It will be observed in Example 1 that the method is capable of piecing together information from multiple documents to develop and verify answer hypotheses and come up with a best answer. It will further be observed in Example 1 that the answer hypotheses developed according to the method typically appear nowhere in the user's question.

3. Glossary

The following terms are intended to have the following general meanings:

Answer: The actual, correct answer to a given question.

Answer hypothesis: A guess at the correct answer to a question, produced by the invention.

Apposition: A grammatical construction in which two (usually adjacent) noun phrases refer to the same thing.

Co-occurrence queries: A secondary query that contains phrases, other than the type phrase, from a user's question.

Corpus, corpora: A body of natural language text to be searched, used by the invention.

Degree of mismatch: A quantified measure of similarity between two phrases.

Document match: The situation where a document satisfies a query.

Domain: A broad area that characterizes an information retrieval task, e.g., general knowledge, medicine, etc.

FSM, finite-state recognizers: A device that receives a string of symbols as input, computes for a finite number of steps, and halts in some configuration signifying that the input has been accepted or else that it has been rejected.

Head noun: The main noun in a noun phrase which may be modified by other words in the phrase. In English, the head noun is usually the rightmost word in the noun phrase.

Indexing: The association of a word with all the different places the word exists in a corpus.

Information retrieval, IR: The accessing and retrieval of stored information, typically from a computer database.

IS-A: The name of a particular lexico-syntactic pattern and linguistic relation. The IS-A relation uses forms of the verb "to be" to denote or assert the identity of or a characteristic of an object. An example of the IS-A relation is seen in the sentence "Brooklyn is a borough of New York City."

Language modeling: Computational modeling of natural language.

Lexico-syntactic: Having to do with words, syntactic patterns, or both. Lexico-syntactic analysis is used in the invention to verify linguistic relations.

List inclusion: The name of a particular lexico-syntactic pattern in which a noun is included in a list of related nouns. Each member of the list has the same relationship as other members of the list to a (primary)

noun outside the list. Typically the nouns of the list indicate objects of the same type, and are separated within the list by commas or semicolons.

Match sentences: Sentences in a document that cause or help cause the document to be retrieved in response to a query. Match sentences contain phrases that conform to the search terms and constraints specified in the query.

Noun phrase: A phrase consisting of a noun, its modifiers (e.g., adjectives and other nouns) and possibly an article.

Noun phrase inclusion: The name of a particular lexicosyntactic pattern.

Operators: kleene-plus, sequence, optionality, union: Functions in the finite-state calculus, used in finite-state recognizers. Optionality means 0 or more instances.

Phrase matching: Finding phrases that satisfy a lexicosyntactic pattern or other linguistic pattern.

Predicate/argument match: Matching a particular lexicosyntactic pattern that represents argument structure. The pattern relates a verb and one or more associated nouns.

Primary documents: Documents from a text corpus that are retrieved in response to a primary query.

Primary query: A query that is based only on phrases from the user's question.

Query: An expression involving words, Boolean operators and proximity constraints that is used by an information retrieval system to search a corpus and return text that matches the expression.

Question: A user's information need, presented to the invention as a natural language question.

Regular expression: A specification of functions in the finitestate calculus.

Relation; phrase relation: A linguistic relation between words or phrases, or between a word or phrase and a certain property (e.g., the property of being a person's name). Put another way, a relation is the way that words work together to create meaning in a phrase or that phrases work together to create meaning in a sentence. A relation has syntactic, semantic, contextual and other aspects. For example, in the sentence, "George Washington was the first president," a relation exists between the phrases "first president" and "George Washington." The syntactic aspects of a relation are used to approximate the relation in some embodiments of the invention.

Scoring: The assignment of values to answer hypotheses, reflecting their likelihood as being the correct answer.

Secondary documents: Documents from a text corpus that are retrieved in response to a secondary query.

Secondary query: A query that is based on an answer hypothesis and that additionally may be based on other phrases.

Shallow syntactic analysis: A syntactic analysis of a phrase that represents gross grammatical structure and not necessarily all the details of the linguistic relationships in the phrase.

Simple noun phrase: A noun phrase that does not contain prepositional phrases or conjunctions.

Tagging: The assignment of part-of-speech categories to words.

Title phrase: A phrase that represents a title (e.g., of a book, film, etc.), which may or may not be a noun phrase, and which typically is indicated typographically (e.g., by enclosing quotes, capital letters, italics, etc.),

Type phrase: A phrase in a question that indicates the nature of the answer, e.g., "What river..." A type phrase can be used to help develop answer hypotheses, because it provides an indication of what the answer should look like. For example, in the question, "Who was the first President of the United States?", the phrase "the first President of the United States" is the type phrase. As another example, in the question, "What river does the Hoover Dam dam?" the type phrase is "river."

Verify, verification: The process of determining whether a relation exists. When a phrase match is verified, this is evidence that the relation holds; that is, text has been found in the corpus that supports the existence of the relation.

PART II. AN EMBODIMENT OF THE METHOD

1. Organization of Part II

Part II of the description sets forth the method of the present invention in one embodiment. Section 2 provides an overview of a system suitable to implement the method. Section 3 outlines the major, high-level steps of the method. Sections 4, 5, 6, and 7 provide detailed breakdowns of four of these top-level steps: question processing, preliminary hypothesis generation, hypothesis verification, and hypothesis ranking. Section 8 describes some variations of the method.

2. System Overview

Fig. 1 illustrates a system suitable to implement the method of the present invention. A user's question and a set of relevant documents is fed to answer extraction subsystem 15 whose software operates on supporting hardware comprising a computing node 4 further comprising a processor 5 and memory 6 coupled to the processor 5. In some embodiments computing node 4 additionally comprises a storage device 8 coupled to processor 5. Answer extraction subsystem 15 is coupled via channel 17 to an information retrieval subsystem 11 that operates on a text corpus 12. In response to questions input by the user at a user interface 7 and a set of relevant documents known to be present in text corpus 12 that is received via channel 16, answer extraction subsystem 15 generates and verifies answer hypotheses. In so doing, answer extraction subsystem 15 can send queries to information retrieval subsystem 11. In response to these queries information retrieval subsystem 11 retrieves documents from text corpus 12 and sends these back to answer extraction subsystem 15. The output of answer extraction subsystem 15 is presented to the user via user interface 7.

Optionally, as further illustrated in Fig. 1, the method of the current invention can be embodied in a system suitable to the two-phase method for answering a natural-language question. Answer extraction subsystem 15 is coupled with primary query construction subsystem 10 via channel 16 and with information retrieval subsystem 11 via channel 17. Answer extraction subsystem 15 receives the output of primary query construction subsystem 10 as its input. This input includes a set of relevant documents and questions input to user interface 7, which are routed to answer extraction subsystem 15 from user interface 7 via primary query construction subsystem 10. Primary query construction subsystem 10 sends queries via channel 14 to information retrieval subsystem 11, which sends back documents or document identifiers retrieved from text corpus 12. Primary query construction subsystem 10 runs on supporting hardware that can comprise node 4 further comprising processor 5 and memory 6 or, alternatively, a separate node comprising its own processor and memory (not shown).

In embodiments that do not include a primary query construction subsystem, the input question and set of documents for answer extraction can be supplied by the user via user interface 7, or, alternatively, retrieved from optional storage device 8.

Certain elements of the system of Fig. 1 will now be described in greater detail.

Answer extraction subsystem (15).

The answer extraction subsystem is a process that executes on one or more computing nodes. By "process" is meant a software entity that comprises one or more threads of execution and possibly one or more input/output streams. By "node" is meant a computer comprising memory and one or more processors. In Fig. 1, answer extraction subsystem 15 executes on a computing node 4 that comprises processor 5 and memory 6.

Computing node, processor, memory, user interface, and optional storage device (4, 5, 6, 7, 8).

The processor 5 and the memory 6 to which it is coupled form the core of a computing node 4 that executes the process that is answer extraction subsystem 15. The computing node 4 can in some embodiments further comprise a storage device 8, such as a hard disk, coupled to the processor 5. The user interface 7 comprises hardware, such as an input keyboard and display screen, and associated software to permit a user to communicate with the answer extraction subsystem. In some embodiments, for example if computing node 4 is a workstation, processor 5, memory 6, user interface 7, and (if present) storage device 8 are integrated into a single computer. In other embodiments, the user interface can run on a processor other than processor 5, or can be implemented on hardware that is at a location remote from processor 5 and memory 6. Persons of skill in the art will appreciate that a vast range of hardware and software can be used to implement node 4, processor 5, memory 6, user interface 7, and optional storage device 8 within the scope of the present invention.

Information retrieval subsystem (11).

The IR subsystem is a process that executes on one or more computing nodes (not shown). Typically the IR subsystem executes on a node or nodes distinct from the node or nodes on which the answer extraction subsystem executes, but this need not be the case. As described more fully below, the IR subsystem is capable of responding to certain primitive queries including Boolean queries with proximity and order constraints, as well as to compound queries formed of these primitives. The IR subsystem 11 is coupled to the text corpus 12 in such a way that the IR subsystem 11, acting in response to an input query, can search the text corpus 12

for documents that match the query.

Text Corpus (12).

The method of the present invention is to be used in conjunction with a text corpus, which is a corpus (body) of information stored in the form of natural language text. Typically the corpus is a collection of articles or documents. Without limitation, kinds of corpora suitable to the invention include, for example, an encyclopedia, a collection of newspaper articles, a set of office correspondence, a set of files containing facts about clients or companies, or a collections of abstracts or articles from scientific or technical journals.

Typically the corpus 12 is stored as a computerized database. The computerized database can be implemented on any suitable hardware; typically such hardware comprises one or more mass storage devices such as hard disks or CD-ROMs.

Channels (14,16, 17).

In a system suitable to the method of the present invention the various processes communicate with one another through communications channels. For example, in Fig. 1, answer extraction subsystem 15 communicates with IR subsystem 11 through channel 17. Where two communicating processes execute on a single node, the communications channels comprise interprocess communication links that also execute on the node. Where two communicating processes execute on different nodes, the communications channel comprises hardware links and software protocols through which the processes can communicate with one another. The communications channels can, for example, be implemented in a local area network or a wide area network and can, for example, incorporate wire, coaxial cable, fiber optic, or wireless hardware links. In one embodiment answer extraction subsystem 15 executes on a workstation at a user's location, IR subsystem 11 executes on a mainframe or server at a remote location (where text corpus 12 is also located), and communications channel 17 comprises a modem link across a wide area network.

It will be appreciated that in some embodiments two or more of the processes described above can be integrated into a single process as modules of that process. For example, the primary query construction subsystem and IR subsystem can be implemented as modules of a single process in some embodiments, or the primary query construction subsystem and an answer extraction subsystem can be implemented as modules of a single process in some embodiments. In such cases the communications channels between the modules can be, for example, shared variables or function or subroutine parameters.

3. Major Steps of the Method

Considered at a high level of abstraction, the method of the present invention proceeds in one embodiment in six major steps, as shown in the flowchart of Fig. 2: (1)input (step 200); (2) question processing (step 220); (3) preliminary hypothesis generation (step 240); (4) hypothesis verification (step 260); (5) hypothesis ranking (step 280); and (6) output (step 290). This section of the description explains how these steps work together in the context of the method as a whole. Subsequent sections examine the steps of question processing, hypothesis generation, hypothesis verification, and hypothesis ranking in greater detail. Still further details of these steps as embodied in the MURAX system are presented in Part III.

In step 200 the answer extraction subsystem accepts as input a natural-language input string (e.g., a question) and a set of documents assumed to contain an appropriate response to the input string (e.g., the answer to the question). The input string is a character string that is input by a user through the user interface and made available to the answer extraction subsystem, either directly or by way of the primary query construction subsystem if such a subsystem is present. The input character string represents a sequence of one or more natural language words. Although the input string is typically in the form of a natural language question, this is not always the case; for example, an input string can be a statement, a paragraph, or even an entire article or other document.

The documents accepted as input in step 200, which will be called the primary documents, are assumed to be present in the text corpus and to contain among them the appropriate response to the input string, e.g., the answer to the question. Typically the primary documents are documents that are retrieved from the text corpus in response to IR queries executed in advance of step 200. The primary documents can also be supplied from other sources. They can be, for example, documents provided by a clipping service, documents produced during litigation in response to a document production request, or documents obtained during a previous information retrieval session using a different corpus.

In step 220 the input string is linguistically analyzed to detect noun phrases in the string. Additional kinds

of phrases can be detected as well. Step 220 is analyzed in more detail in section 4 below. The result of step 220 is an analysis of the input string.

Step 220 can be performed by the answer extraction subsystem after receiving the input string. Alternatively, it can be performed in advance of step 200 by another process. In this case a completed analysis of the input string is supplied to the answer extraction subsystem as additional input along with the input string and primary documents in step 200.

In step 240 the answer extraction subsystem generates preliminary hypotheses from the analyzed input string and the primary documents. In one embodiment preliminary hypothesis generation comprises locating match sentences in the documents, scoring these match sentences, extracting noun phrases from the match sentences and from adjacent sentences in the primary documents, and scoring these noun phrases to generate a ranked list of preliminary hypotheses. Step 240 is analyzed in more detail in section 5 below. The result of step 240 is a set of selected hypotheses.

In step 260 the answer extraction subsystem verifies hypotheses from the set of selected hypotheses produced in step 240. In one embodiment hypothesis verification comprises lexico-syntactic analysis, template matching, and optional linking of equivalent hypotheses. The template matching procedure can incorporate the construction and execution of secondary queries. Step 260 is described in more detail in section 5 below. The result of step 260 is a set of verification evidence and optional linking information for each hypothesis.

In step 280 the answer extraction subsystem performs hypothesis ranking according to a scoring scheme. The goal of this step is to rank highest the answer hypothesis or hypotheses most likely to be responsive to the input string. Step 280 is analyzed in more detail in section 5 below. The result of step 280 is an ordered list of answer hypotheses.

In step 290 the answer extraction subsystem outputs a subset of the ordered list of answer hypotheses produced in step 280. The subset can be output directly to the user via the user interface. Alternatively or additionally it can be stored in a storage device for later use, or made available for further processing. In some embodiments one or more answer hypotheses can be highlighted in the documents in which they appear for ease of reference. In other words, the answer extraction subsystem tells the user what it thinks the answer is and why. In some embodiments output to the user can be done in an interactive fashion, for example, by permitting the user to issue commands to the system to display answer hypotheses only, to display answer hypotheses in the context of the documents in which they appear, etc.

4. Question Processing

This section examines step 220, question processing, in detail. In one embodiment question processing comprises shallow linguistic analysis of the natural language input string, which is typically (but not necessarily) in the form of a question.

Fig. 3 flowcharts the component steps of question processing in an embodiment of the method that uses shallow linguistic analysis. In step 300 the input string is analyzed to determine what part of speech each word of the string is. Each word of the string is tagged to indicate whether it is a noun, verb, adjective, etc. Tagging can be accomplished, for example, by a tagger that uses a hidden Markov model. The result produced by step 300 is a tagged input string.

After step 300 is complete, steps 310, 315, and 320 are performed either in series or, as illustrated in Fig. 3, in parallel. In step 310, which comprises component steps 311 and 312, the tagged input string is analyzed to detect noun phrases. In step 315 the tagged input string is further analyzed to detect main verbs. In step 320 the input string is analyzed to detect title phrases. Each of steps 310,315, and 320 will now be further described.

In step 310 noun phrases are detected. A noun phrase is a word sequences that consists of a noun, its modifiers such as adjectives and other nouns, and possibly a definite or indefinite article. Step 310 comprises steps 311 and 312. In step 311 the tagged input string is analyzed to detect the maximal-length simple noun phrases that it contains. This can be done, for example, using a finite state machine (FSM) noun phrase recognizer. See J.E. Hopcroft and J.D. Ullman, Introduction to Automata Theory, Languages, and Computation (Addison-Wesley, 1979). In step 312 case-splitting is performed wherein maximal length noun phrases detected in step 311 can be split into short noun phrases according to initial word capitalization. In particular, a sequence of words within a noun phrase that is itself a noun phrase and whose words all have their first letters capitalized is split from other words in the noun phrase. For example, if in step 311 the noun phrase "New York City borough" is detected, then in step 312 it can be split as

New York City | borough

because the initial letters of the words of "New York City" are capitalized while the initial letter of "borough" is not. Once this split is determined, for purposes of further processing the noun phrase "New York City borough"

is discarded and is replaced by the two shorter noun phrases "New York City" and "borough." As another example, if in step 311 the noun phrase "last Apache warrior chief" is detected, then in step 312 this phrase is split according to its initial capitalization as

last | Apache | warrior chief

and the noun phrases "Apache" and "warrior chief" are used in further processing while the original noun phrase "last Apache warrior chief" is discarded. The adjective "last," which after the splitting is no longer part of a noun phrase, is likewise discarded.

In step 315 main verbs are detected. Main verbs are any words that are tagged in step 300 as verbs and that are not auxiliary verbs. Typically there is one main verb in the input string, but there can also be none, or two or more.

In step 320 title phrases are found in the input string. Title phrases are word sequences that represent titles of books, motion pictures, or other works. They can in some cases include or even be the same as noun phrases detected in step 310. In other cases they can include verbs or other parts of speech that are not noun phrase components. Title phrases can be recognized, for example, according to typographical indications such as being enclosed in quotation marks or typeset in italics. Thus the following input string fragments would be recognized as title phrases:

"Gone With the Wind"

Gone With the Wind

Typically the input string contains zero or one title phrases, but it is also possible for it to contain two or more.

In step 330 the results of steps 310, 315, and 320 are stored. The stored results represent the completed analysis of the input string. The results can be stored, for example, in a list of 3-tuples, one 3-tuple for each noun phrase, main verb, and title phrase detected during steps 310, 315, and 320. Each 3-tuple is an ordered list of the form (i, phrase-type, text), where i is a unique index number associated with the phrase, such as its position (first, second, third ...) in the list; phrase-type indicates the type of phrase (noun phrase, main verb, or title phrase); and text is a string that contains the text of the phrase itself. Thus for example if the noun phrases "Abraham Lincoln" and "theater," the main verb "shoot," and the title phrase "Reconsidering the Civil War" are detected in the input string, the list of results, expressed here in Lisp syntax, can be of the form

((1 'NP "Abraham Lincoln") (2 'NP "theater")
(3 'V "shoot") (4 'TP "Reconsidering the Civil War"))

The list can be stored in the processor's memory. Alternatively or additionally it can be stored in a storage device coupled to the processor. It will be appreciated that although step 330 is illustrated in Fig. 3 as a step that is performed separately from and subsequently to steps 310, 315, and 320, in some embodiments an empty list is created as part of step 330 at the outset, prior to the execution of steps 310,315, and 320, and thereafter is filled in incrementally during the processing of the steps 310, 315, and 320, so that upon completion of steps 310, 315, and 320, step 330 is effectively completed as well.

5. Preliminary Hypothesis Generation

This section examines step 240, preliminary hypothesis generation, in detail. Fig. 4 flowcharts the component steps of preliminary hypothesis generation in one embodiment of the method. In step 241 match sentences are located in the primary documents. Sentences are found in the primary documents that contain phrases from the input string or parts of such phrases. The sentences can be found by automatically constructing IR queries based on the phrases detected during question processing step 220 and executing these queries on the primary documents. (It is assumed that at least some of the primary documents match the queries.) Alternatively, if question processing is carried out ahead of time by a primary query construction process, match sentences can also be detected by the primary query construction process and made available to answer extraction subsystem as additional input in step 200.

In step 245 match sentences are heuristically scored. Match sentences having the highest scores are retained for further processing. The scores of the match sentences and their locations within their respective primary documents are recorded as well. The number of match sentences retained at this step is determined by a parameter set by default or in advance by the user.

The heuristic scoring of step 245 attempts to score each match sentence according to its degree of commonality with the input string. Points are awarded to each sentence based on criteria such as the number of phrases of the input string that the match sentence contains, the lengths of these phrases, and whether all or only some of the words of these phrases are found in the corresponding phrases of the match sentence. Other strategies can be used in other embodiments. For example, all sentences can be assigned the same score (e.g., 1) in some embodiments.

In step 250 the match sentences retained for further processing in step 245 are analyzed to detect phrases

they contain. The match sentences are analyzed in substantially the same manner as the input string is analyzed in step 220 above. The detected phrases typically comprise noun phrases and can further comprise title phrases or other kinds of phrases. The phrases detected in the match sentences are called preliminary hypotheses.

Each preliminary hypothesis is associated with the document in which it was detected. For example, if the phrase "Robert Kennedy" is detected three times in document doc1 and twice in document doc2, then the answer extraction subsystem will develop two distinct preliminary hypotheses in step 250, which can be represented as:

("Robert Kennedy" doc1)
("Robert Kennedy" doc2)

In other embodiments, preliminary hypotheses that comprise the same phrase-that is, preliminary hypotheses that include the same exact sequences of words--can be distinguished based on subdivisions of documents, such as sections, paragraphs, or sentences. For example, in an embodiment where preliminary hypotheses that come from different paragraphs are considered to be distinct, then if in document doc1 "Robert Kennedy" appears twice in the first paragraph and once in the tenth paragraph, and in document doc2 "Robert Kennedy" appears once in the fourteenth paragraph, three preliminary hypotheses are generated, which can be represented as:

("Robert Kennedy" doc1 paragraph 1)
("Robert Kennedy" doc1 paragraph 10)
("Robert Kennedy" doc2 paragraph 14)

In still other embodiments preliminary hypotheses can be treated as equivalent regardless of their textual source.

Each preliminary hypothesis is also associated with the particular match sentences that gave rise to it. If a hypothesis is a phrase of a match sentence it is associated with that match sentence. The association of hypotheses with match sentences implies a further association between hypotheses and the match sentence scores determined in step 245. Each preliminary hypothesis can be assigned a preliminary score based on the scores of the match sentences with which it is associated. The preliminary score can be, for example, the sum of the match sentence scores.

In step 255 the preliminary hypotheses developed in step 250 are heuristically scored and are ranked in order from highest score to lowest. The heuristic scoring of step 255 is based on the match sentence scores determined in step 245. In one embodiment the match sentence scores associated with each unique preliminary hypothesis—that is, each unique sequence of words that makes up one or more document-specific preliminary hypotheses—are summed to generate the score for step 255. For example, suppose that:

1) Document doc10 contains the phrase "Robert Kennedy" twice, in match sentences whose scores are 120 and 60 respectively.

2) Document doc30 contains the phrase "John Kennedy" four times, in match sentences whose scores are 10, 10, 40, and 50 respectively.

3) Document doc200 contains the phrase "Robert Kennedy" once, in a match sentence whose score is 50.

In this case, the scores assigned to the three preliminary hypotheses are as follows:

Table 1:

Preliminary Scores Example	
Preliminary Hypothesis	Score
("Rober Kennedy" doc 10)	120+60+50 = 230
("John Kennedy" doc 30)	10+10+40+50= 110
("Rober Kennedy" doc200)	120+60+50 = 230

In other words, the two preliminary hypotheses ("Robert Kennedy" doc10) and ("Robert Kennedy" doc200) receive identical scores based on the sum of the scores of all match sentences containing the unique phrase "Robert Kennedy" regardless of the textual source of such sentences. The preliminary hypothesis "John Kennedy" is treated likewise; however, since it appears only in sentences that received low match scores, it receives a lower overall score. It will be observed that in an embodiment in which all match sentences are assigned equal scores of 1 in step 245, the score assigned in step 255 is a simple frequency count. Thus in the

example, the hypotheses ("Robert Kennedy" doc10) and ("Robert Kennedy doc200) receive scores of 3 and the hypothesis ("John Kennedy" doc30) receives a score of 4 in such an embodiment.

In step 259 the N highest-ranked preliminary hypotheses of the ranked list are selected for verification in step 260. N is a parameter determined, for example, by default or by the user in advance. For example, N can be set equal to a fixed, predetermined number, such as 30

6. Hypothesis Verification

This section examines step 260, hypothesis verification, in detail. In one embodiment hypothesis verification comprises using lexicosyntactic analysis to verify linguistic relations for the selected hypotheses produced by step 240, and linking equivalent hypotheses to produce a set of linking information.

6.1 Lexico-Syntactic Analysis

Hypotheses are verified in step 260 through lexico-syntactic analysis. Lexico-syntactic analysis comprises analysis of linguistic relations implied by lexico-syntactic patterns in the input string, constructing or generating match templates based on these relations, instantiating the templates using particular hypotheses, and then attempting to match the instantiated templates, that is, to find primary or secondary documents that contain text in which a hypothesis occurs in the context of a template.

Lexico-syntactic analysis is illustrated by the following example. Suppose that the input string is the question, "Who succeeded Shastri as Indian prime minister?", and that there are numerous primary documents, most of which contain the term "prime minister." These documents concern various prime ministers throughout history and throughout the world, for example, Winston Churchill, Golda Meir, Brian Mulroney, etc. Included among the primary documents is an article that contains the sentence, "Indira Gandhi was prime minister..." Suppose further that also among the documents of the corpus, either among the primary documents or elsewhere, is an article that contains the sentence "Indira Gandhi succeeded Shastri..." but that does not refer to Indira Gandhi as "prime minister." The answer to the question can be constructed only by reference to both articles together. In lexico-syntactic analysis for this example, the answer extraction subsystem constructs or generates one or more appropriate match templates based on word patterns in the input string. To do so it uses part-of-speech information about these words, such as that generated during question processing. In the question "Who succeeded Shastri..." the word "succeeded" is a verb. The answer extraction subsystem recognizes this and further recognizes that questions that begin with "who" often call for answers that are persons' names. Accordingly, the answer extraction subsystem constructs a template of the form "X succeeded Shastri" and tries to find a document containing a phrase that matches this template. Here, the template is matched by the phrase "Gandhi succeeded Shastri" in the article that contains the sentence "Indira Gandhi succeeded Shastri..." The answer extraction subsystem can also, for example, construct a template of the form "Shastri was succeeded by X," which would be matched by a document containing the phrase "Shastri was succeeded by Indira Gandhi...."

The use of the template technique gives the method of the present invention more robust performance that would be had if a simple cooccurrence approach were used for verification. In the example, the sentence "Indira Gandhi succeeded Shastri..." provides much stronger support for the answer hypothesis "Indira Gandhi" than the mere co-occurrence of the names "Shastri" and "Indira Gandhi" in a sentence or document. Furthermore, the template technique can be used in a domain-independent manner; that is, there is no need for humans to pre-analyze the primary documents or the text corpus. For example, in order to answer the question above the answer extraction subsystem does not need to refer to a table of prime ministers.

6.2 Component Steps of Hypothesis Verification

Fig. 5 flowcharts the component steps of hypothesis verification step 260 in one embodiment of the method. In step 261 linguistic relations to be verified are determined using lexico-syntactic analysis of the input string. A loop is executed for each such linguistic relation. In step 262 at least one template is constructed for the linguistic relation under consideration. At this point a loop is executed for each of the answer hypotheses. In step 263 templates are filled in using the hypothesis under consideration. In step 264 an attempt is made to verify the linguistic relation for the hypothesis under consideration by matching the filled-in templates in the primary documents. If sufficient matches are found, this attempt succeeds and processing continues with the next hypothesis. However, if no matches or only a few are found, the attempt fails and processing continues at step 265. In step 265 at least one secondary query is constructed and executed. If documents are retrieved by the IR subsystem in response to the secondary query or queries, then in step 266 an attempt is made to

verify the linguistic relation for the hypothesis under consideration through template matching using the retrieved documents. After all linguistic relations and hypotheses have been processed, in step 270 equivalent hypotheses are optionally linked.

The procedure of step 260 can be expressed as pseudocode, for example as follows:

```
DETERMINE linguistic relations in the input string;

/* Outer loop over L linguistic relations LR_i in the input string. */
FOR i = 1 to L
  {CONSTRUCT templates T_i,j based on LR_i;

  /* Inner loops over M templates T_i,j and N hypotheses H_k.
     The nesting order of these loops can be reversed, depending on the particular
     implementation. */
     FOR j = 1 to M
     FOR k = 1 to N
     {{FILL IN template T_i,j using hypothesis H_k;
      ATTEMPT TO VERIFY LR_i by seeking matches to the
      filled-in templates in the primary documents;
      IF (attempt does not succeed)
         {CONSTRUCT AND EXECUTE secondary queries;
         IF (secondary documents are retrieved from text
             corpus in response to secondary queries)
             {ATTEMPT TO VERIFY LR_i by
              seeking matches to the filled-in templates in
              secondary documents
             }
         }
     }}
  }
LINK equivalent hypotheses H_k;
```

Each of the steps 261, 262, 263, 264, 265, 266, and 270 will now be more fully described.

6.3 Linguistic Relations

In step 261 the input string is analyzed to determine the linguistic relations that are to be verified in subsequent steps. The analysis is done by seeking certain lexico-syntactic patterns in the input string based on the analysis of the input string that was performed during question processing. The patterns are assumed to correspond to particular linguistic relations. For example, if the user's question begins, "Who was the Chief Justice who presided over...", then the answer to the question can be in the form "The Chief Justice was..." Thus the linguistic IS-A relation applies. A single lexico-syntactic pattern can imply multiple relations.

In an embodiment of the invention in which the linguistic relations IS-A, list inclusion, apposition, and noun phrase inclusion are among the relations that can be verified, all these relations can be used to verify an answer hypothesis as an instance of a type phrase in the input string. A type phrase is a phrase in a question that

indicates the nature of the appropriate answer to the question. In questions of the form, "What is X?", or, "Who is X?", the type phrase is X. To determine the type phrase for such a question, the answer extraction subsystem seeks one of the words "who" or "what," followed by the verb "to be," followed by a noun phrase. That noun phrase is taken to be the type phrase. Other lexico-syntactic patterns can also give rise to type phrases. For example, in the question, "What river does the Hoover Dam dam?" the type phrase is "river." The pattern here is the word "what" followed immediately by a noun phrase.

Some user questions have no type phrases associated with them. For example, the question, "Who won twenty Oscars?" has no type phrase associated with it. This is because the word "won" does not establish a type phrase relation. Also, even for questions having type phrases, it can be beneficial to seek additional kinds of relations. Some of the other relations that can be sought are transitive verb relations and identification as a proper names.

Lexico-syntactic patterns in the input string can be seen as clues used by the answer extraction subsystem to help it determine what kinds of answer hypotheses are likely to be the most appropriate responses to the input string. For example, if the input string is a question, the interrogative words and syntactic structure of the question provide clues about the kind of answer that is expected. A question that asks "Who..." or "Whose..." often calls for a person's name as an appropriate answer. A question that begins "What X...", "Which X...", "Who was the X...", or "What is the X...", where X is a type phrase, often calls for an answer that is an instance of X. A "Where..." question typically calls for a location or place name, and a "When..." question for a time or date. A "How many..." question calls for a number. Put another way, lexico-syntactic patterns inform the answer extraction subsystem about the kind of evidence that must be gathered in order to verify the hypotheses.

6.4 Templates

For each of the linguistic relations detected in step 261, step 262 is performed. In step 262 the answer extraction subsystem constructs or otherwise generates one or more templates for the linguistic relation under consideration. A template is a lexico-syntactic pattern, typically not one present in the input string, that when instantiated (filled in) can be searched for in the documents. When an instantiated template is matched by the text of a document, this is evidence that the linguistic relation is satisfied.

Templates are so called because they contain one or more placeholders that can be filled in with hypotheses. When a hypothesis appears with other words in a sentence or phrase of a document in such a way as to satisfy the lexico-syntactic pattern of a template, that hypothesis is assumed to relate to the other words according to the linguistic relation to which the template corresponds. In this case the template is said to be matched for the given hypothesis. Thus a linguistic relation can be verified by filling in the template with the hypothesis and matching the filled-in template to text in a document.

For a question that has a type phrase, the answer extraction subsystem generates templates that incorporate the head word of the type phrase. A head word is the most significant word of the type phrase. In English, the head word is generally the noun that appears rightmost in the type phrase. For example, in the type phrase, "Chief Justice," the head word is "Justice." Templates based on head words are constructed by substituting the head word from the type phrase and a placeholder for the hypothesis in a lexico-syntactic pattern corresponding to one of the linguistic relations IS-A, list inclusion, apposition, or noun phrase inclusion. For example, if the hypotheses about who might be the Chief Justice are "Oswald," "Warren," and "Kennedy," and the IS-A relation is being considered, then a template can be generated of the form "Justice was X." This template will match phrases such as, "Justice was Warren" or "Justice was Kennedy." If the list inclusion relation is considered, templates can be generated of the form "Justices X, Y,..." to match phrases such as "Chief Justices Warren, Burger, Rehnquist...." If the apposition relation is considered, templates can be generated of the form "X, NP(Justice)" to match phrases such as "Earl Warren, Chief Justice...." Here, NP(Justice) means a noun phrase containing "Justice."

When a question has no type phrase associated with it, lexicosyntactic analysis is performed, although not of the type-phrase variety. For example, the template "X (won, wins) twenty Oscars" can be generated and matched for the input string "Who won twenty Oscars?" The linguistic relation is the transitive verb relation implied by the verb "win" and the direct object "twenty Oscars."

In step 263, which is executed for each hypothesis, the templates are filled in with the hypothesis under consideration. That is, the hypothesis under consideration is substituted for a placeholder or placeholders in the template.

6.5 Matching Templates Against Primary Documents

In step 264 an attempt is made to verify the linguistic relation under consideration for the hypothesis under

consideration in the context of the primary documents. This is done by matching the filled-in templates generated in step 263 against the primary documents. In other words, sentences in which the hypothesis appears in the context of a template are sought in the primary documents. Any such sentences found are retained in association with the hypothesis as verification evidence for use in later processing steps.

For example, if the template is "NP(Justice) (is,was) X" and the hypothesis is "Earl Warren," the filled-in template is "NP(Justice) (is,was) Earl Warren," and documents containing sentences such as "At that time the Chief Justice was Earl Warren..." are potential matches. As another example, if the template is "X succeeded Shastri" and the hypothesis is "Indira Gandhi," the filled-in template is "Indira Gandhi succeeded Shastri." The answer extraction subsystem seeks one or more primary documents that contain sentences conforming to this filled-in template, for example, "Indira Gandhi succeeded Shastri...."

The testing of step 264 is carried out using only the primary documents. If sufficient template matches are found among the primary documents, then the linguistic relation is considered verified. In this case it is unnecessary to run secondary queries and steps 265 and 266 are skipped for this linguistic relation and hypothesis.

6.6 Secondary Queries

If few or no template matches are found for in step 264, then in steps 265 and 266 an attempt is made to verify the linguistic relation under consideration for the hypothesis under consideration in the text corpus at large. In step 265 secondary queries are constructed and executed. Documents retrieved in response to secondary queries are called secondary documents. If secondary documents are successfully retrieved in step 265, then in step 266 the filled-in templates generated in step 263 are matched against the secondary documents in the same manner as they were matched against the primary documents in step 264.

Secondary queries are IR queries designed to find documents that potentially contain template matches--that is, documents in which the hypothesis under consideration appears in the context of the template, thus indicating that it satisfies the linguistic relation to which the template corresponds. Accordingly, a secondary query typically seeks the cooccurrence of an answer hypothesis and one or more phrases of the input string, such as the type phrase (if one is present), a main verb or verb phrase, or a title phrase. Portions of such phrases can be used as well, for example, the head word of the type phrase or the head word of the hypothesis. The documents retrieved in response to secondary queries contain sentences that include both the input string phrase or phrases and the hypothesis.

Secondary queries are formulated in a query language that expresses Boolean, proximity, and ordering or sequence relationships between search terms (individual words or other queries) in a form understandable by the IR subsystem. If a secondary query retrieves no documents or a number of documents below a predetermined threshold, it can be broadened and retried. Broadening can be accomplished, for example, by relaxing order or proximity constraints.

Secondary queries are performed in step 265 if in step 264 no template matches are found among the primary documents or only a few template matches are found. In the latter case, there is a possibility that the matches are false matches, so that additional supporting evidence is needed for the hypothesis under consideration. (It will be observed that secondary queries themselves can generate false matches.)

A single secondary query can sometimes serve to gather evidence for multiple relations. For example, a secondary query that seeks the cooccurrence of a hypothesis with the head noun of a type phrase can serve to retrieve documents from which any of the relations IS-A, apposition, list inclusion, or noun phrase inclusion can be satisfied. In some embodiments each secondary query is recorded after it is constructed or executed so that redundant secondary queries can be avoided.

If secondary documents are retrieved in response to the secondary queries in step 265, step 266 is performed. In step 266 the secondary documents retrieved in step 265 are analyzed to determine whether their match sentences do in fact satisfy the template. For example, if the template is of the form "X to be NP," where X is the hypothesis and NP is a noun phrase, then a secondary query that seeks the co-occurrence of the hypothesis and noun phrase can be performed in step 265. If this query is successful, then in step 266 it is determined whether the hypothesis and noun phrase are properly connected by the verb "to be" in one or more of the match sentences in which they appear. This is done by applying the template to the match sentences, that is, by seeking the filled-in template in the match sentences. Any match sentences determined in step 266 to satisfy the template are retained in association with the hypothesis as verification evidence for use in later processing steps.

Secondary queries can be of particular importance when the number of primary documents is limited. This can be the case, for example, in an embodiment in which primary documents are provided by a primary query construction subsystem that is designed to retain only a specified number of documents thought to be most

relevant (e.g., the 15 top-ranked documents). In such an embodiment it is possible for a document that contains the answer to the user's question to be found by the primary query construction subsystem but then discarded because it is not considered sufficiently relevant. For example, suppose that in the example above concerning Indira Gandhi the document containing the sentence "Indira Gandhi succeeded Shastri..." was ranked by the primary query construction subsystem as being too insignificant to be retained. Therefore this document would not be supplied to the answer extraction subsystem as a primary document, and so in step 263 no primary documents would be found that contain matches to the template, "X succeeded Shastri." In other words, as of step 264, the only point in Mrs. Gandhi's favor would be the same as that for Mr. Churchill or Ms. Meir, namely, that all were "prime ministers." Accordingly, in step 265 secondary queries are generated for Churchill and Shastri, Gandhi and Shastri, etc. That is to say, the answer extraction subsystem looks specifically for the names of various prime ministers who are answer hypotheses in the same context as the name "Shastri." These secondary queries turn up the document that was previously discarded. In step 266 the answer extraction subsystem determines that this document contains the match sentence "Indira Gandhi succeeded Shastri...," which supports Mrs. Gandhi as the best answer hypothesis.

6.7 Linking Equivalent Hypotheses

Once template matching is complete for all relations and hypotheses, different hypotheses that are likely to refer to the same answer can be linked in step 270. Hypotheses that differ from one another in wording or that come from different textual sources (in this embodiment, from different documents) but that probably refer to the same thing are caused to be treated as equivalents--in other words, they are linked together. Step 270 is optional and is omitted in some embodiments.

As an example, consider the hypotheses

("Robert Kennedy" doc1)
("Kennedy" doc1)
("Robert F. Kennedy" doc2)
("Robert Kennedy" doc20)
("Robert M. Kennedy" doc37)

The problem of linking is to determine whether these hypotheses refer to the same person, for example, the Robert F. Kennedy who was at one time attorney general of the United States under President John F. Kennedy. If document doc1 primarily concerns John F Kennedy, it is not immediately apparent whether the "Kennedy" in doc1 refers to John or Robert. Similarly, it is not immediately apparent whether the "Robert Kennedy" in doc1 is the same as that in doc20, or whether either of these refers to the "Robert F. Kennedy" of doc2 or the "Robert M. Kennedy of doc37.

Linking can be accomplished using a wide variety of heuristics and approaches. Some examples illustrate. One heuristic is to treat as equivalent hypotheses that are different forms of the same proper name if these hypotheses occur close to one another in a single document. For example, "Kennedy" is likely to refer to the same person as "Robert Kennedy" if both appear in the same paragraph or document, especially if "Robert Kennedy" appears first in the text and "Kennedy" appears thereafter. Another heuristic is to treat as equivalent all instances of a proper noun in all documents when the proper noun is a single word that is never modified by other words, for example, "Pythagoras" or "Stonehenge."

Linking strategies can be adapted to the text corpus used in a particular embodiment. For example, if the corpus comprises encyclopedia articles, a hypothesis that appears in an article title is likely to refer to the article's subject. In particular, an encyclopedia article that contains birth and death dates and numerous personal pronouns is likely to be a person's biography and to have as its title the person's name, and a hypothesis that is a form of the person's name is likely to refer to that person. For example, an encyclopedia article on John F. Kennedy is likely to be entitled "John F. Kennedy" and often to refer to its subject as "Kennedy."

The result of step 270 is a set of linking information for the hypotheses, or, viewed differently, a smaller set of hypotheses. The linking information tells which hypotheses are to be considered equivalent to which other hypotheses. For example, suppose once again that the unlinked hypotheses are

("Robert Kennedy" doc1)
("Kennedy" doc1)
("Robert F. Kennedy" doc2)
("Robert Kennedy" doc20)
("Robert M. Kennedy" doc37)

Suppose further that as a result of step 270 the first three of these hypotheses are linked to one another and the last two are linked to one another. Then the result of step 270 is linking information that can be expressed as

(link1 ("Robert Kennedy" doc1)
("Kennedy" doc1)
("Robert F. Kennedy" doc2))
(link2 ("Robert Kennedy" doc20)
("Robert M. Kennedy" doc37))

Alternatively it can be said that there are now two hypotheses, "Robert F. Kennedy" and "Robert M. Kennedy" instead of the original five hypotheses listed above.

7. Hypothesis Ranking

This section describes step 280, hypothesis ranking, in detail. The goal of hypothesis ranking is to determine which answer hypotheses are most relevant or responsive to the input string. If the input string is a question, for example, the purpose of hypothesis ranking is to decide which hypothesis is likely to be the best answer to the question.

Hypothesis ranking can be done by assigning a score to each hypothesis and ordering the hypotheses according to the scores thus assigned. The highest-ranked hypotheses are considered the best answers. Any number of scoring methods can be used. Scoring can be based in whole or in part on the verification evidence collected in step 260.

Fig. 6 illustrates the component steps of hypothesis ranking. In step 281 scores are computed for the hypotheses. In step 283 scores of linked hypotheses are merged. In step 285 the hypotheses are ranked based on their scores. In step 287 results are organized for presentation. Each of these steps will now be more fully described.

7.1 Scoring

In step 281 scores are assigned to the (unlinked) hypotheses. In one embodiment each hypothesis score is based on three criteria. The first criterion is verification evidence obtained through template matching in primary and secondary documents in step 260. The second criterion is cooccurrence of the hypothesis with phrases of the input string in primary and secondary documents, regardless of whether templates were matched. The third criterion is the preliminary hypothesis score developed in step 240, which is based on the scores of the primary document match sentences from which the hypothesis derives.

In one specific embodiment the scoring of an answer hypothesis can be performed as follows: The first criterion--that is, scoring based on verification evidence--is applied when the input string is a question having a type phrase. For each answer hypothesis, all sentences in primary or secondary documents in which a template based on the type phrase was matched for the hypothesis are taken into account, and the maximum number of type phrase words matched in any such sentence is determined. This number becomes the basis for scoring according to the first criterion. For instance, if the type phrase is "big blue truck," and one hypothesis is "truck" and another is "blue truck," then the latter hypothesis is to be preferred because it matches more type phrase words. Accordingly it is assigned a higher score.

Consider an example that will be called Example 2. Suppose that the input string is the question, "What Chief Justice led the commission that investigated the assassination of John F. Kennedy?" The type phrase here is "Chief Justice" and the head word is "Justice." Suppose further that there are four answer hypotheses, namely, "Oswald," "Earl Warren," "Kennedy," and "Warren Earl"

Table 2:

Scoring Measures for Example 2	
Hypothesis	Type phrases words matches
Oswald	0
Earl Warren	2
Kennedy	1
Warren Earl	2

Table 2 shows that the hypothesis "Oswald" (which turns out to refer to Lee Harvey Oswald) has no tem-

plate matches associated with it. The hypothesis "Earl Warren" has associated with it sentences for which templates based on the type phrase were matched; the maximum number of type phrase words matched within any such sentence is 2. "Kennedy" appears in sentences for which a maximum of 1 word of the type phrase was matched. Finally, "Warren Earl" (which turns out to refer to Chief Justice Warren Earl Burger) gives rise to template matches in which the maximum number of words matched from the type phrase is 2.

The results in Table 2 can further be interpreted as follows: Although it is possible that sentences in the primary and secondary documents contain both of the words "Oswald" and "Justice" (that is, the hypothesis and the head word of the type phrase, respectively), none of these sentences satisfies a template for any of the desired linguistic relations (e.g., the template "X (is,was) NP(Justice)" for the IS-A relation). There are sentences that contain both "Earl Warren" and "Justice" and that match templates for the IS-A, apposition, list inclusion, or noun phrase inclusion relations; the maximum number of words of the type phrase "Chief Justice" matched in any of these is 2. For "Kennedy," there are sentences that includes both the word "Kennedy" and the word "Justice." Of these sentences, at least one gives rise to a template match, which in this example turns out to be a false match: The template "Kennedy (is,was) NP(Justice)" matches a sentence that contains the word sequence "A high priority for Kennedy was racial justice." Since the phrase "racial justice" shares only one word with the type phrase "Chief Justice," the maximum number of matching words for is 1 for this hypothesis. Finally, "Warren Earl" (Burger), who was Chief Justice after Earl Warren, appears in some sentences that match templates. The maximum number of words matched in any of these sentences is 2 (the two words "Chief Justice").

Thus in this example, the first scoring criterion leads to a tie. "Earl Warren" and "Warren Earl" tie for highest score with 2 type phrase words matched apiece. "Kennedy" is next with 1 type phrase word matched, and "Oswald" scores the lowest with 0 words matched.

Next, the second scoring criterion--that is, co-occurrence of the hypothesis with phrases of the input string--is applied to break the tie among hypotheses. (It can also be applied if there is no type phrase in the question.) The second criterion is an empirical one that is designed to increase the scores of hypotheses appearing in documents that are on the topic of the user's original question. The score for this criterion is based on the maximum number of noun phrases of the user's question that appear in any of the primary documents that contain the hypothesis. In Example 2, the hypotheses "Earl Warren" and "Warren Earl" can be distinguished based on the second criterion, because only the name "Earl Warren" appears frequently in conjunction with the other phrases of the question such as "John F. Kennedy," "commission," and "assassination." This is because Earl Warren figures prominently in the history of the investigation of the Kennedy assassination. The hypothesis "Warren Earl" scores lower, because the documents in which Chief Justice Warren Earl Burger's name appears are for the most part not related to the Kennedy assassination.

In this specific embodiment, information from multiple documents is pieced together at the time the second criterion is applied. An instance of this is seen in Example 1 above ("What Pulitzer Prize-winning novelist ran for mayor of New York City?"). In Example 1, not all the information needed to determine the best answer hypothesis appears in any one document. Information from two documents must be integrated. In particular, the information that "Norman Mailer" is a "novelist," which appears in a secondary document, must be integrated with the information that "Norman Mailer" was a mayoral candidate, which appears in a primary document.

The noun phrases of the original question of Example 1 are "Pulitzer Prize," "winning novelist," "mayor," and "New York City." The type phrase is "winning novelist" and its head noun is "novelist." The answer hypotheses are "Edith Wharton," "William Faulkner," and "Norman Mailer." Suppose that each of the hypotheses matches the type phrase "winning novelist" with a maximum of one word, "novelist," so that the first criterion does not distinguish among the hypotheses. Suppose further that the hypotheses co-occur with the noun phrases of the question as follows in the primary documents (X indicates a co-occurrence):

Table 3:

Input phrase co-occurences for Example 1				
Hypothesis	"Pulitzer Prize"	"winning novelist"	"mayor"	"Nex York City"
Edith Wharton	X	X		X
William Faulkner	X	X		X
Norman Mailer	X		X	X

The hypothesis "Edith Wharton" appears in at least one primary document that also includes the phrases "Pu-

litzer Prize," "novelist," and "New York City." In other words, the maximum number of phrases from the question that appear in any one primary document along with the hypothesis "Edith Wharton" is three. Similarly, the hypothesis "William Faulkner" appears in at least one primary document with "Pulitzer Prize," "novelist," and "New York City." Again the maximum number of input string phrases that appear in conjunction with "William Faulkner" is three. "Norman Mailer," who according to the primary documents was a "writer" but not a "novelist," appears in at least one document that contains all three of the phrases "Pulitzer Prize," "mayor," and "New York City."

At this point information from secondary documents is integrated. In this embodiment, if a hypothesis is verified in a secondary document as being an instance of the type phrase, then the hypothesis is scored according to the second criterion as though the type phrase had appeared in a primary document that contains the hypothesis and a maximum number of input string phrases. Thus in Example 1, because the hypothesis "Norman Mailer" appears in a secondary document as an instance of the type phrase "novelist," the hypothesis "Norman Mailer" is scored according to the second criterion as though the type phrase "novelist" had appeared in one of the same primary documents that also contains "Norman Mailer," "Pulitzer Prize," "New York City," and "mayor." In other words, because the answer extraction subsystem knows from a secondary query phrase that "Norman Mailer" was indeed a "novelist," this information is read into Table 3, and the box that indicates a co-occurrence of "Norman Mailer" and "winning novelist" is filled in as in Table 4.

Table 4:

Expanded input phrase co-occurences for Example 1				
Hypothesis	"Pulitzer Prize"	"winning novelist"	"mayor"	"New York City"
Edith Wharton	X	X		X
William Faulkner	X	X		X
Norman Mailer	X	X	X	X

Then this augmented version of the table is used to apply the second scoring criterion. The hypothesis "Norman Mailer" is now treated as appearing in conjunction with four rather than three input string phrases, namely, "Pulitzer Prize," "winning novelist," "mayor," and "New York City." Accordingly this hypothesis scores higher on the second criterion than the other two hypotheses.

The third scoring criterion—the preliminary scores that were obtained during the step of generating hypotheses, that is, step 240--is applied in the specific embodiment as a tie-breaker if the first two criteria do not suffice to distinguish among the hypotheses. The hypothesis that was assigned the highest preliminary score is chosen over others that scored equally by the first two criteria.

The use of multiple criteria for the determination of the final score leads to more robust system performance than would be obtained with single criteria used in isolation. Through the use of multiple scoring criteria, the system can determine one or two best possible answer hypotheses in the context of articles that are most likely to be relevant.

7.2 Merging Scores

Step 283 is performed if linking was performed in step 270. In step 283 the linking information generated in step 270 is used to merge the scores of linked hypotheses. Each hypothesis in a set of linked hypotheses receives a merged score that is computed using the evidence gathered for all the hypotheses of the set. For example, the merged score can be based on the sum of the individual scores, or based on the number of unique templates (regardless of how instantiated) matched for the hypotheses of the set. More sophisticated schemes can be used.

As an example, suppose that there are six hypotheses and two links as follows:

```
(link1("Robert Kennedy" doc1)
        ("Kennedy" doc1)
        ("Robert F. Kennedy" doc2))
(link2 ("Robert Kennedy" doc20)
        ("Robert M. Kennedy" doc37))
("John Kennedy" doc30)
```

The first three hypotheses are linked together, and the fourth and fifth hypotheses are linked together. The

last hypothesis, which is the phrase "John Kennedy" from document doc30, is not linked to any other hypothesis. Suppose further that the scores of the individual hypotheses are as follows:

Hypothesis	Score
("Robert Kennedy" doc1)	50
("Kennedy" doc1)	20
("Robert F. Kennedy" doc2)	80
("Robert Kennedy" doc20)	40
("Robert M. Kennedy" doc37)	10
("John Kennedy" doc30)	100

If the scores for linked hypotheses are merged through summation, the resulting scores are as follows:

Hypothesis	Merged Score
("Robert Kennedy" doc1)	50 + 20 + 80 = 150
("Kennedy" doc1)	50 + 20 + 80 = 150
("Robert F. Kennedy" doc2)	50 + 20 + 80 = 150
("Robert Kennedy" doc20)	40 + 10 = 50
("Robert M. Kennedy" doc37)	40 + 10 = 50
("John Kennedy" doc30)	100

In this example each hypothesis receives a merged score based on its own individual score and the individual scores of all hypotheses to which it is linked. For instance, the first hypothesis, ("Robert Kennedy" doc1), receives a score that is the combined individual scores of the first three hypotheses. The last hypothesis, ("John Kennedy" doc30), is not linked to any other and so retains its previous score.

7.3 Ranking Hypotheses and Organizing Results

In step 285 the hypotheses are ranked according to their scores from highest to lowest. This step can be accomplished by a straightforward sorting procedure. If linking is performed in step 270 then equivalent linked hypotheses are eliminated prior to sorting. For each set of linked hypotheses, a representative member is selected, for example, the hypothesis containing the greatest number of words. Thus "Robert F. Kennedy" would be selected over "Robert Kennedy" or "Kennedy." The representative member is used in the ranking.

As an example, suppose once again that the linked hypotheses are

(link1 ("Robert Kennedy" doc1)
("Kennedy" doc1)
("Robert F. Kennedy" doc2))
(link2 ("Robert Kennedy" doc20)
("Robert M. Kennedy" doc37))
("John Kennedy" doc30)

and that the hypotheses

("Robert F. Kennedy" doc2)
("Robert M. Kennedy" doc20)

are selected to represent the hypotheses of link1 and link2 respectively. If the scores assigned to these hypotheses are, respectively, 150 and 50, and the score assigned to the unlinked hypothesis ("John Kennedy" doc30) is 100, then the ranked hypotheses produced by step 285 are

1. ("Robert F. Kennedy" doc2)
2. ("John Kennedy" doc30)
3. ("Robert M. Kennedy" doc20)

In step 287 the ranked hypotheses are organized into results suitable for output. In one embodiment in which results are to be presented to the user, the highest-ranked answer hypothesis is selected for presentation. This hypothesis is highlighted in the contexts in which it appears in primary and secondary documents, for example by displaying the document titles and the match sentences that confirm the linguistic relations implied by the user's question. The hypothesis can be emphasized through underlining or a distinctive font. Phrases of the input string that appear in context with the hypothesis can likewise be emphasized. Additionally, the answer extraction subsystem can provide further information about verification, linking, and scoring. In short, the answer extraction subsystem provides results that tell the user what the best answer hypothesis is, where it occurs in the documents, and why this answer was selected. The second and third-ranked hypotheses can be also presented, for example by themselves without the supporting information.

In some embodiments, step 287 incorporates selecting which documents to present from numerous documents containing the best answer hypothesis. For example, if many documents match the best answer hypothesis, the one or two documents having the shortest matching sentences containing the hypothesis can be selected for presentation.

8. Alternative Embodiments

The foregoing embodiment of the method is only one possible embodiment among many. Moreover, variations of the method fall within the scope of the invention. A few alternative embodiments will now be briefly described. Some of these involve alternative techniques for executing certain method steps. Others involve omitting certain method steps. Further alternatives will be apparent to those of skill in the art.

8.1 Alternative Techniques for Method Steps

Alternative techniques can be used to carry out the input string analysis in question processing step 220. For example, a parser based on a context-free grammar can be used. Case-splitting of noun phrases need not be performed. Cues other than or in addition to text formatting can be used to detect title phrases. Most generally, any technique can be used that converts the natural language input string into phrases—that is, contiguous, relatively short sequences of words. The types of phrases detected need not be limited to noun phrases, title phrases, and verb phrases.

Step 240 can likewise be carried out using alternative techniques for analyzing match sentences. Also, different approaches to the scoring and ranking of preliminary hypotheses can be used. Sentences that appear nearby (e.g., adjacent to or in the same paragraph as) the match sentences in the primary documents can be analyzed in addition to the match sentences themselves in order to detect phrases to be used as preliminary hypotheses.

Step 260 can be carried out using alternative techniques for analyzing the input question, generating templates, analyzing match sentences, and generating secondary queries. Linguistic relations and corresponding lexico-syntactic patterns other than those described above can be used both to analyze the input string and to generate additional templates. For example, relative clauses such as "Who is the king who sat at the water's edge....?" can be used to generate the templates "X is a king" and "X sat at the water's edge," which can each be used for hypothesis verification. Sentences nearby the match sentences can be used for verification, in addition to the match sentences themselves. Secondary queries can be used to generate additional hypotheses beyond those obtained from the primary documents. Secondary queries can be nested and run iteratively to generate still better answer hypotheses. Multiple nested secondary queries can be generated and executed to an arbitrary level of nesting. If a secondary query is generated to verify a linguistic relation and the secondary query itself leads to a new relation to be verified, a tertiary query can be run; if the tertiary query calls for additional verification, a fourth-level query can be run; and so forth. Techniques other than template matching can be used for detecting lexico-syntactic patterns; for example a rule set can be used. Sophisticated linking techniques, for example techniques involving heuristics and rule sets, can be used.

Also in step 260, non-lexico-syntactic approaches to hypothesis verification can be used in place of or in addition to lexico-syntactic hypothesis verification. Such approaches can include the construction and execution of secondary queries that are not based on templates. For example, if the input question calls for a person's name as an answer, and the text corpus comprises encyclopedia articles, the following is one possible non-lexico-syntactic technique to verify that a given hypothesis is in fact a person's name. A secondary query that comprises the hypothesis with no other search terms is constructed and executed. Among the retrieved secondary documents, the answer extraction seeks a document that has the hypothesis as its title. If such a document is found, its content is examined to see, for example, whether it mentions birth and death dates and whether it contains a large number of personal pronouns such as "he" or "she." If so, the document is most probably

a biography of the person whose name is the document's title. Therefore the hypothesis is most probably a person's name.

Another example of a non-lexico-syntactic verification technique that can be used in step 260 involves queries that can be called secondary co-occurrence queries. Such queries are performed during verification to seek co-occurrences of an answer hypothesis with any input string phrases that do not appear in any recognized relations and are thus not incorporated into any templates. In the absence of template matches, the fact that the answer hypothesis co-occurs with an input string phrase provides at least weak evidence for the hypothesis.

More generally, instead of or in addition to lexico-syntactic pattern analysis, hypothesis verification in step 260 can incorporate more sophisticated or powerful linguistic analysis, such as can be provided by various natural language parsing techniques, including techniques that employ semantic information.

Steps 240 and 260 can incorporate the use of reference documents. For example, a thesaurus can be made available to the answer extraction subsystem, either as part of the subsystem itself or as an external resource connected to the subsystem by channels. The thesaurus can be used to generate synonyms and hyponyms (e.g., for type phrase word replacement) that can in turn be used for generating hypotheses, matching sentences during verification, or both. A table of given names and surnames can be used to help verify that hypotheses are persons' names. A gazetteer can be used to provide information about place names.

Step 280 can be performed using any number of different scoring techniques. For example, in embodiments where the three scoring criteria presented above (that is, verification evidence obtained through template matching, co-occurrence of the hypothesis with phrases of the input string regardless of whether templates were matched, and preliminary hypothesis score) are used, the first criterion can take into account matches of any kind of template used for verification, not just templates based on a type phrase. The fact that a particular template is matched against a document for a particular hypothesis is assigned importance and adds to the score of the hypothesis. The number of times that a template is matched for the hypothesis can contribute to the score, as can the number of times the hypothesis appears together with the head noun of the type phrases. Different templates can also be assigned different importances according to the linguistic relations to which they correspond. The second and third criteria can be folded in with the first criterion in a weighted sum, rather than being used only to break ties as in the specific embodiment above. For the second criterion, other sources of information can be used in lieu of or in addition to type phrase matching information when creating an augmented table such as Table 4. Still more sophisticated strategies, including strategies not limited to the three criteria described above, can be used. Such strategies can include the use of semantic information.

8.2 Omitting Certain Method Steps

The method of the present invention is a method for organizing information retrieval based on the content of a set of documents. The method generates answer hypotheses based on text found in the documents of the set. In some embodiments, such as those described up to this point, information retrieval is additionally based on an input string. If there is an input string, the answer hypotheses can include phrases or words not present in the input string. In some embodiments, such as those described up to this point, the answer hypotheses are verified and can be ranked based on their verification evidence. If verification is performed, a text corpus can be used to provide verification information not present in the set of documents. The documents can be present in the text corpus, but need not be.

In the alternative embodiments that will now be described, certain steps described for the embodiments described up to this point are omitted. In one such alternative embodiment the step of hypothesis verification is skipped entirely. The input, question processing, and preliminary hypothesis generation, and output steps are performed. The hypothesis ranking step collapses into the steps of ranking and selecting of preliminary hypotheses, and the hypotheses' preliminary scores become their final scores. Presented with an input string and a set of documents, this embodiment uses text of the documents to develop a set of hypotheses relevant to or responsive to the input string.

In another such alternative embodiment, no input string is supplied and the steps of question processing and hypothesis verification both are skipped. Presented with a set of documents, this embodiment attempts to develop a hypothesis about what the documents have in common. It can, for example, search for all the noun phrases in all the documents of the set and determine which noun phrase occurs most often.

In still other such alternative embodiments the step of hypothesis verification can be included but without secondary queries, or without linking of equivalent hypotheses.

PART III. MURAX

1. Organization of Part III

Part III of the description focuses on MURAX, a system that embodies the method of the present invention in the context of a larger twophase method for answering a natural-language question. MURAX provides a robust linguistic approach for answering natural-language questions about specific facts. It derives its answers to questions from a database comprising an on-line encyclopedia. MURAX has been implemented and tested using a hardware configuration in which a Sun SparcStation workstation runs both the primary query construction and answer extraction subsystems, and the information retrieval subsystem and its associated corpus are located remotely from the Sun workstation and are accessed by the workstation via a network.

The rest of Part III is organized as follows: In Section 2 the motivation for MURAX's question-answering task is given, along with a description of the kind of questions that are its concern and their characteristics. Section 3 describes the system components. These include the encyclopedia and the IR system for accessing it. Shallow linguistic analysis is done using a part-of-speech tagger and finite-state recognizers for matching lexico-syntactic patterns.

Section 4 describes the analysis of a question by considering an example and illustrates the system output. Analysis proceeds in two phases. The first, primary query construction, finds articles that are relevant to the question. The second phase (called answer extraction) analyzes these articles to find noun phrases (called answer hypotheses) that are likely to be the answer.

Both phases require searching the encyclopedia. Queries made during the first phase are called primary queries, and only involve phrases from the question. The second phase creates secondary queries which MURAX generates to verify specific phrase relations. Secondary queries involve both answer hypotheses and phrases from the question.

Section 5 explains the primary query construction phase, which is carried out in MURAX according to the method of co-pending application entitled METHOD FOR COMPUTERIZED INFORMATION RETRIEVAL USING SHALLOW LINGUISTIC ANALYSIS as incorporated hereinabove by reference. Section 6 explains the answer extraction phase, which is carried out in MURAX according to the method of the present invention. Section 7 presents a performance evaluation of MURAX.

2. Task Selection

MURAX's task is to answer certain general-knowledge questions using a text corpus that comprises an encyclopedia. There are several reasons for choosing this task. Robust analysis is needed for the task because the encyclopedia is composed of a significant quantity of unrestricted text. General knowledge is a broad domain, which means that it is impractical to manually provide detailed lexical or semantic information for the words of the vocabulary (the encyclopedia contains over 100,000 word stems).

MURAX demonstrates that shallow syntactic analysis can be used to practical advantage in broad domains, where the types of relations and objects involved are not known in advance and may differ for each new question. Its analysis capitalizes on the information available in a question and profits from treating the encyclopedia as a lexical resource. MURAX also demonstrates that natural language analysis can add to the quality of the information retrieval process, providing text to the user that confirms phrase relations and not just word matches.

MURAX uses questions that have definite answers. This means that its performance can be evaluated in a straightforward way by using a set of questions and correct answers. Given a correct noun phrase answer, it is generally easy to judge whether a noun phrase hypothesized by the system is correct or not. Thus relevance judgements are simplified, and if one correct hypothesis is considered as good as any other, recall measurements are not required and performance can be considered simply as the percentage of correctly hypothesized answers.

2.1 Question Characteristics

MURAX operates on closed-class questions. A closed-class question is a direct question whose answer is assumed to lie in a set of objects and is expressible as a noun phrase. Put another way, a closedclass question is a question, stated in natural language, that assumes some definite answer typified by a noun phrase rather than by a procedural answer. Examples of closed-class questions are given below in Table 5.

Table 5

	Example questions
1.	What U.S. city is at the junction of the Allegheny and Monongahela rivers?
2.	Who wrote "Across the River and into the Trees"?
3.	Who married actress Nancy Davis?
4.	What's the capital of the Netherlands?
5.	Who was the last of the Apache warrior chiefs?
6.	What chief justice headed the commission that declared: "Lee Harvey Oswald... acted alone."?
7.	What famed falls are split in two by Goat Island?
8.	What is November's birthstone?
9.	Who's won the most Oscars for costume design?
10.	What is the state flower of Alaska?

(These Trivial Pursuit questions are Copyright Horn Abbott Ltd. Trivial Pursuit is a Registered Trademark of Horn Abbot Ltd.)

The questions in Table 5 appear in the general-knowledge "Trivial Pursuit" game and typify the form of question that is the concern of the MURAX task. These questions have the virtue of being created independently of the retrieval task (that is, they are unbiased) and have a consistent and simple stylized form. Yet they are flexible in their expressive power.

The interrogative words that introduce a question are an important source of information. They indicate particular expectations about the answer and some of these are illustrated below in Table 6.

Table 6:

Question Words and Expectations

Who/Whose: *Person*

What/Which: *Thing, Person, Location*

Where: *Location*

When: *Time*

How Many: *Number*

Notable omissions here are the words *why* and *how*. Questions that begin with "why" or "how" typically expect a procedural answer rather than a noun phrase answer (e.g., "How do you make a loaf of bread?").

The expectations established by the interrogative words can be used to filter various answer hypotheses. The answers to questions beginning with the word "who" are likely to be people's names. This fact can be used to advantage because various heuristics can be applied to verify whether a noun phrase is a person's name.

A question introduced by "what" may or may not refer to a person; however, other characteristics can be exploited. Consider the following sentence fragments, where *NP* symbolizes a noun phrase: "What is the *NP*..." and "What *NP*...". The noun phrase at the start of such questions is called the question's *type phrase* and it indicates what type of thing the answer is. The encyclopedia can be searched to try to find evidence that an answer hypothesis is an instance of the type phrase (details are presented in section 6.1.1 below). The verbs in a question are also a useful source of information as they express a relation that exists between the answer and other phrases in the question.

The answer hypotheses for "Where..." questions are likely to be locations, which often appear with locative prepositions or as arguments to verbs of motion. Questions of the form "When..." often expect answer hypotheses that are dates or times and the expectation of questions beginning "How many..." are numeric expres-

sions.

3. Components

An on-line version of Grolier's Academic American Encyclopedia (The Academic American Encyclopedia, Danbury, Connecticut: Grolier Electronic Publishing, 1990) serves as the text corpus for MURAX. The online encyclopedia contains approximately 27,000 articles, which are accessed via the Text Database (D.R Cutting, J. Pedersen, and P.-K. Halvorsen, "An object-oriented architecture for text retrieval," in Conference Proceedings of RIAO '91, Intelligent Text and Image Handling, Barcelona, Spain, April 1991, pages 285-298), which is a flexible platform for the development of retrieval system prototypes and is structured so that additional functional components, e.g., search strategies and text taggers (see D. Cutting, J. Kupiec, J. Pedersen, and P. Sibun, "A practical part-of-speech tagger," in Proceedings of the Third Conference on Applied Natural Language Processing, Trento, Italy, April 1992, ACL) can be easily integrated.

The components responsible for linguistic analysis in MURAX are a part-of-speech tagger and a lexico-syntactic pattern matcher. The tagger is based on a hidden Markov model (HMM). HMM's are probabilistic and their parameters can be estimated by training on a sample of ordinary untagged text. Once trained, the Viterbi algorithm is used for tagging.

To assess performance, an HMM tagger (J.M. Kupiec, "Robust partof-speech tagging using a hidden Markov model," Computer Speech and Language, 6:225-242, 1992) was trained on the untagged words of half of the Brown corpus (W.N. Francis and F. Kutera, Frequency Analysis of English Usage, Houghton Mifflin, 1982) and then tested against the manually assigned tags of the other half. This gave an overall error rate of 4% (corresponding to an error rate of 11.2% on words that can assume more than one part-of-speech category). The percentage of tagger errors that affect correct recognition of noun phrases is much lower than 4%. The tagger uses both suffix information and local context to predict the categories of words for which it has no lexicon entries.

The HMM used for tagging the encyclopedia text was also trained using the encyclopedia. A benefit of such training is that the tagger can adapt to certain characteristics of the domain. An observation in this regard was made with the word "I". The text of the encyclopedia is written in an impersonal style and the word is most often used in phrases like "King George I" and "World War I". The tagger trained on encyclopedia text assigned "I" appropriately (as a proper noun) whereas the tagger trained on the Brown corpus (a mixture of different kinds of text) assigned such instances as a pronoun.

Given a sentence of text, the tagger produces a sequence of pairs of words with associated part-of-speech categories. These enable phrase recognition to be done. Phrases are specified by regular expressions in the finite-state calculus (Hopcroft and Ullman, supra). Noun phrases are identified solely by part-of-speech categories, but more generally categories and words are used to define lexico-syntactic patterns against which text is matched.

Initially, only simple noun phrases are identified because they are recognized with the greatest reliability. Analysis involving prepositional phrases or other coordination is applied subsequently as part of more detailed matching procedures.

Word-initial capitalization turns out to be useful for splitting noun phrases appropriately. Thus "New York City borough" is split into "New York City" and "borough". Such case-splitting improves the efficiency of Boolean query construction, because queries based on case-split noun phrases tend to generate phrase matches directly This reduces the need to drop several words successively from a noun phrase in order to produce a match.

3.1 Title Phrases

A title phrase is a multi-word phrase that is the title of a film, book, play, etc. It is usefully treated as a single unit in MURAX. A title phrase need not be a noun phrase. For example, the title "Play Misty for Me" contains the verb "play".

Title phrases are readily identified when marked typographically by enclosing quotes or italics. However, title phrases may be marked only by word-initial-capitalized letters. Furthermore, some title phrase words, such as short function words, need not be capitalized. Thus, the correct extent of the title phrase can be ambiguous and alternative possibilities must be accommodated. In MURAX the most likely alternative is chosen after phrase matching has been done and the alternatives compared, based on the matches and frequency of the alternative interpretations.

4. Operational Overview

This section presents an overview of the two-phase operation of the MURAX system, along with an example of the output of the system. The context is the example question presented Table 7 below.

Table 7:

Example Question and Component Noun Phrases	
"Who was the Pulitzer Prizewinning novelist that ran for mayor of New York City?"	
Pulitzer Prize	winning novelist
mayor	New York City

4.1 Primary Query Construction

In its first phase of processing MURAX embodies the method of co-pending application entitled METHOD FOR COMPUTERIZED INFORMATION RETRIEVAL USING SHALLOW LINGUISTIC ANALYSIS as incorporated hereinabove by reference. Simple noun phrases and main verbs are first extracted from the input question, as illustrated in Table 7. These phrases are used in a query construction/refinement procedure that forms Boolean queries with associated proximity and order constraints (section 5). The queries are used to search the encyclopedia to find a list of relevant articles.

The relevant articles are heuristically scored according to the degree and number of matches with the phrases of the input question. Matching head words in a noun phrase receive double the score of other matching words in a phrase. Words with matching stems but incompatible part-of-speech categories are given minimal scores.

After scoring, the relevant articles are ranked according to their scores. A subset of the ranked articles is made available for the second phrase of processing. MURAX assumes that these articles, which are called the primary documents, contain the answer to the user's question.

4.2 Answer Extraction

In its second phase of processing MURAX embodies the method of the present invention.

Answer extraction begins by finding all simple noun phrases contained in the match sentences of the primary documents, that is, in those sentences that caused or helped to cause the primary documents to be retrieved The match sentences correspond to one or more phrases of the input question. Each noun phrase found becomes an answer hypothesis that is distinguished by its component words and the article and match sentence in which it occurs.

Answer hypotheses are scored on a per-article basis according to the sum of the scores of the articles in which they occur. The purpose of this is to minimize the probability of overlooking the correct answer hypothesis if a subsequent non-exhaustive search is performed using the hypotheses.

For each answer hypothesis the system tries to verify phrase relations implied by the question. For the question in Table 7, it will be observed that the answer is likely to be a person (indicated by "who"). The type phrase indicates that the answer is preferably a "Pulitzer Prize winning novelist", or at least a "novelist" as indicated by the head noun of the type phrase. The relative pronoun indicates that the answer also "ran for mayor of New York City". Phrase matching procedures (detailed in section 6) perform the verification using the answer hypotheses and the primary documents.

Verification is not limited to primary documents. It can happen that a pertinent phrase relation is not present in the primary documents although it can be confirmed elsewhere in the encyclopedia. This is because too few words are involved in the relation in comparison to other phrase matches, so the appropriate sentence does not rank high enough to be in the selected primary documents. It is also possible that appropriate verification information is not expressed in any primary document and depends only on the answer hypothesis. This is the case with one heuristic that MURAX uses to attempt to verify that a noun phrase represents a person's name. The heuristic involves looking for an article that has the noun phrase in its title; thus if the article does not share any phrases with the question, it would not be part of any primary document.

Secondary queries are used as an alternative means to confirm phrase relations. A secondary query may consist of solely an answer hypothesis (as for the heuristic just mentioned) or it may also include other phrases

from the input question, such as the type phrase of the question. To find out whether an answer hypothesis is a "novelist", for example, a query comprising the answer hypothesis and the word "novelist" is constructed and executed to yield a list of relevant articles. Documents whose match sentences contain co-occurrences of an answer hypothesis and a question phrase are called secondary documents. MURAX analyzes secondary documents using lexico-syntactic patterns to see if answer hypotheses can be validated as instances of the type phrase.

4.3 System Output

For the question given in Table 7 MURAX produces the output shown in Table 8.

Table 8:

Output for Example Question

The best matching phrase for this question is: **Mailer, Norman**

The following documents were most relevant:

Document Title: **Mailer, Norman**

Relevant Text:

- "The Armies of the Night (1968), a personal narrative of the 1967 peace march on the Pentagon, won **Mailer** the **Pulitzer Prize** and the National Book Award."

- "In 1969 **Mailer** ran unsuccessfully as an independent candidate for **mayor** of **New York City**."

Document Title: novel

Relevant Text:

- "Among contemporary American novelists, Saul Bellow, John Dos Passos, John Hawkes, Joseph Heller, **Norman Mailer**, Bernard Malamud, Thomas Pynchon, and J.D. Salinger have reached wide audiences."

Next best: Edith Wharton, William Faulkner

The presentation here is different from that of prior art IR systems. Answer hypotheses are shown to the user to focus his or her attention on likely answers and how they relate to other phrases in the question. The text presented is not necessarily from documents that have high similarity scores, but rather from documents that confirm phrase relations that lend evidence for an answer. This behavior is readily understood by users, even though they have not been involved in the tedious intermediate work done by the system.

In Table 8, the first two sentences are from primary documents. The last sentence confirming Norman Mailer as a novelist is a from a secondary document. The sentence was confirmed by a lexico-syntactic pattern which identifies the answer hypothesis as being in a list-inclusion relationship with the type phrase.

MURAX's approach contrasts with the alternative, prior art approach of vector-space search. Vector-space search using full-length documents is not as well suited as MURAX's approach to the task that MURAX performs. For the example question, a vector-space search was done using a typical similarity measure and the bag of content words of the question. The most relevant document (about Norman Mailer) was ranked 37th.

Somewhat better results can be expected if sentence or paragraph level matching is done. However, the resulting text matches do not have the benefit of being correlated in terms of a particular answer, and they muddle information for different answer hypotheses.

5. Primary Query Construction

This section describes in further detail how MURAX performs primary query construction. In accordance with the method of the co-pending application entitled METHOD FOR COMPUTERIZED INFORMATION RETRIEVAL USING SHALLOW LINGUISTIC ANALYSIS as incorporated hereinabove by reference, MURAX translates phrases from a question into Boolean queries with proximity and order constraints. These queries are passed to an IR system which searches the encyclopedia and returns a list of matching documents (or *hits*). It is assumed that the IR system can process the following kinds of primitive queries, as well as compound queries constructed of these primitives:

1. The Boolean AND of terms, denoted here as:
   $[term_1 term_2 ... term_n]$
2. Proximity of a strict sequence of terms, separated by up to p other terms denoted here as:
   $<p\ term_1, term_2 ... term_n$
3. Proximity of an unordered list of terms, separated by up to p other terms denoted here as:
   $(p\ term_1, term_2 ... term_n)$

The overall process is illustrated with an example question, which will be called Example 3 below: "Who shot President Lincoln?"

The question is first tagged and the noun phrases and main verbs are found. In Example 3 the only noun phrase is *President Lincoln* and the main verb is *shot*. Boolean terms are next constructed from the phrases. At the outset a strict ordering is imposed on the component words of phrases. For Example 3, the first query is:

<0 president lincoln>

The IR system receives this Boolean query and searches for documents that match. Depending on the number of hits, new Boolean queries may be generated with the purposes of:

1. Refining the ranking of the documents.
2. Reducing the number of hits (narrowing).
3. Increasing the number of hits (broadening).

5.1 Narrowing

Narrowing is performed to attempt to reduce the number of hits. It can also serve to refine the ranking of the documents. One way that MURAX accomplishes narrowing is by constructing and executing new queries that use title phrases rather than noun phrases or that include additional search terms such as main verbs. Including the main verb in the query for Example 3 gives:

[<0 president lincoln > shot]

Another way that MURAX achieves narrowing is by reducing the cooccurrence scope of terms in the query. This constrains phrases to be closer together, and thus indirectly increases the probability of the phrases' being in some syntactic relation with each other. A sequence of queries of increasingly narrow scope is made, the sequence being carried out until the number of hits falls below some predetermined threshold. A narrowed version of the main verb query for Example 3 is shown below:

(10 <0 president lincoln> shot)

5.2 Broadening

Broadening is performed to attempt to increase the number of hits. It is achieved in three ways in MURAX:

1. By increasing the co-occurrence scope of words within phrases, while at the same time dropping the requirement for strict ordering of the words. In Example 3, for instance, the broadened query (5 President Lincoln) would match the phrase "President Abraham Lincoln" whereas the first query <0 President Lincoln> would not. MURAX performs a sequence of queries of increasingly broad scope. The sequence continues until some threshold on either the proximity or the resulting number of hits is reached.
2. By dropping one or more whole phrases from the Boolean query. Query terms, each corresponding to a phrase, are dropped to get more hits. It is efficient to drop them in an order that corresponds to decreasing number of overall occurrences in the encyclopedia.
3. By dropping one or more words from within multiple-word phrases in a query to produce a query that

is composed of sub-phrases of the original. To increase the number of hits in Example 3, *president* could be dropped, and so might *Lincoln.*

5.3 Control Strategy

An initial Boolean query that comprises all the noun phrases derived from the user's question is first constructed and executed. Broadening and narrowing are then performed. The following partial order can be used:

1. Co-occurrence scope is increased before terms are dropped.
2. Individual phrases are dropped from a query before two phrases are dropped.
3. Higher frequency phrases are dropped before lower frequency ones.
4. Title phrases are tried before any of their component noun phrases.
5. Complete phrases are used before their subphrases.

The iterative process of broadening and narrowing terminates when either a threshold on the number of hits has been reached, or no further useful queries can be made. Upon termination the hits are ranked. In practice it is not necessary to provide elaborate ranking criteria and documents are ranked simply by the number of terms they have in common with the user's question.

6. Answer Extraction

This section describes in further detail how MURAX performs answer extraction in accordance with the method of the present invention. MURAX finds the most likely answer hypotheses from the relevant sentences in the various hits found during the primary query construction phase. Phrase matching operations are conducted first, followed by a procedure for constructing secondary queries to get secondary documents. Generally, several hypotheses can represent the same answer, so they must be linked together and their various phrase matches combined. The hypotheses are then ranked in order of likelihood.

6.1 Phrase Matching

Phrase matching is done with lexico-syntactic patterns which are described using regular expressions. The expressions are translated into finite-state recognizers, which are determinized and minimized (Hopcroft and Ullman, supra) so that matching is done efficiently and without backtracking. Recognizers are applied to match sentences in primary and secondary documents, and the longest possible phrase match is recorded.

An example pattern and text match is shown in Table 9. For convenience, copies of expressions can be included by naming them in other expressions. In Table 9, the expression NP1 refers to a noun phrase whose pattern is defined elsewhere.

Table 9:

Example Pattern and Text Match	
Regular Expression Operators:	
+	One or more instances
?	Zero or one instances
{...}	sequence of instances
(...)	inclusive-or of instances
Lexico-Syntactic pattern:	
{	NP1 (are were include {such as})
+	{NP2,}
?	NP3 ? {and NP4}}
Example match	
"Countries such as Egypt, Sudan, and Israel..."	
NP1 NP2 NP2 NP4	

For robustness, MURAX layers phrase matching on top of cooccurrence matching. Accordingly, if the input is not a question (or is a question beginning with "how" or "why"), MURAX provides output that is typical of co-occurrence based search methods.

MURAX's large corpus mitigates some of the problems inherent in using simple language modelling. For example, in a document match, a relation may not be verified because such verification requires more sophisticated analysis than is feasible with a simple finite-state grammar. However, the relation may be expressed in several places in the encyclopedia and thus more simply in some places, which improves the chances that it can be verified. As another example, simple phrase matching can lead to spurious matches. Other things being equal, an answer hypothesis having more instances of a match is preferred. However, it is less likely that spurious matches for an answer hypothesis will occur for several different phrase relations that appear in different places in the encyclopedia. Thus the large corpus tends to prevent spurious-match errors from propagate far enough to cause an erroneous answer.

6.1.1 Verifying Type Phrases

The following relations are used by MURAX to try to verify answer hypotheses as instances of type phrases:

Apposition. This is exemplified by the match between the type phrase of the following question and the match sentence below it:

"Who was the last Anglo-Saxon king of England?"

1) "The last Anglo-Saxon king of England, Harold, b.c. 1022, was defeated and killed at..."

The IS-A Relation This is demonstrated by the following match sentence:

2) "Saint Edward the Confessor, b. between 1002 and 1005, d. Jan. 5, 1066, was the next to last Anglo-Saxon king of England (1042-66)."

List Inclusion. Lists are often used to enumerate objects of the same type. Examples are seen in Tables 8 and 9.

Noun Phrase Inclusion . Type phrases are often related to answer hypotheses by being included in them. In the question and corresponding match sentence shown below, the type phrase *river* is in the same noun phrase as the answer hypothesis *Colorado River:*

"What river does the Hoover Dam dam?"

"...the Hoover Dam, on the Colorado River..."

6.1.2 Predicate/Argument Match

This operation associates answer hypotheses and other noun phrases in a match sentence that satisfy a verb relation implied in a question. Verbs are simply assumed to be monotransitive and patterns accounting for active and passive alternation are applied. This is illustrated by the question and match sentence shown below:

"Who succeeded Shastri as prime minister?"

"...Shastri was succeeded by Indira Gandhi as Indian prime minister..."

6.1.3 Minimum Mismatch

For reliable identification, simple noun phrases are extracted from match sentences of primary documents. For the question in Table 7, the phrase "mayor of New York City" is first considered as two simpler and independent noun phrases. Exact matching of the overall noun phrase is done after all match sentences are found.

When comparing type phrases with answer hypotheses, the minimum degree of mismatch is considered best. This is illustrated by considering the first question in section 6.1.1 above and the associated match sentences (1) and (2). Both answer hypotheses, "Harold" and "Saint Edward the Confessor," match equally well with the type phrase "last Anglo-Saxon king of England". However, "Harold" is (correctly) preferred because the match is exact, whereas a longer match is involved for "Saint Edward the Confessor" (namely, he was the "next to last Anglo-Saxon king of England").

6.1.4 Person Verification

Confirming an answer hypothesis as a person's name is important. In the encyclopedia, a reliable property of peoples' names is that they have word-initial capital letters. This simple consideration significantly reduces the number of answer hypotheses that require further consideration.

Many different multi-national names are present in the encyclopedia, and exhaustive manual enumeration is impractical. However, there are indirect clues that can be used to verify that an answer hypothesis is a name. Encyclopedia articles about a person generally have the person's name as the title and typically mention birth and/or death dates, which can easily be identified, at the beginning of the article. Usually in such articles there is also a higher percentage of words that are male or female pronouns than in other articles.

Accordingly, to try to confirm an answer hypothesis as a person's name, MURAX makes a secondary query to see if the answer hypothesis is present as an article title, and then decides whether the article is about a person. This heuristic is simple yet robust.

6.2 Secondary Queries

Match sentences in secondary documents are a supplementary means of confirming phrase relations. Secondary documents are found via secondary queries that are constructed by MURAX and passed to the IR system. Broadening is applied as necessary to secondary queries, but terms are never dropped because they are required in the resulting matches. For person verification, the secondary query contains only the answer hypothesis. However, verification of other relations requires other question phrases to be included in the secondary query, as will now be described more fully. In all these secondary queries, answer hypotheses are included verbatim.

6.2.1 Type Phrase Queries

When trying to verify a type phrase, MURAX includes only the head word of the type phrase in the secondary query. This provides the minimal necessary constraint on document matches. The detailed matching of all words in the type phrase is done by considering the degree of mismatch with the type phrase (as in section 6.1.3 above).

When the type phrase cannot be matched against an answer hypothesis using any lexico-syntactic pattern, the fact of their cooccurrence in a sentence is still recorded, as it may serve as a means of ranking alternative hypotheses in the absence of any better information (the relation can still be implied by the document match even if it cannot be inferred from the simple matching operations that MURAX uses for verification).

6.2.2 Co-Occurrence Queries

It is expedient to include other question phrases in secondary queries. As mentioned above in section 4.2, a relevant phrase match can be missed because the primary document in which it occurs has too low a score in comparison to other primary documents. Creating secondary queries with individual question phrases allows the relevant phrase match to be found by bringing documents excluded during primary query construction back into consideration.

Secondary queries are also used to find co-occurrences of answer hypotheses and question phrases that extend beyond the context of a single sentence. This can be useful for ranking alternative answer hypotheses in the absence of other differentiating phrase matches. It is illustrated in the following question and primary document match sentences:

"What film pits Humphrey Bogart against gangsters in the Florida Keys?"

"...Bacall and Bogart became a famous romantic couple in films such as The Big Sleep (1946) and Key Large (1948). "

"Some of his most popular films were The Maltese Falcon (1941); Casablanca (1942), with Ingrid Bergman; The Big Sleep (1946) costarring his wife, Lauren Bacall; The Treasure of Sierra Madre (1948);..."

Secondary co-occurrence queries determine that the answer hypothesis Key Largo co-occurs with Florida Keys, but the other "film" hypotheses do not; thus in the absence of stronger evidence to the contrary, *Key Largo* receives a preference.

6.3 Equivalent Hypotheses

Answer hypotheses are identified by the article and match sentence in which they occur. Generally, the same answer is expressed by several hypotheses. This can happen, for example, when an answer hypothesis refers to a person, because often different word sequences are used to refer to the same person. For example, *President Kennedy, John F. Kennedy, President John F. Kennedy* all refer to the same person. In certain articles of the encyclopedia, so does *Kennedy*; the usage can be disambiguated in such articles by reference to the articles' titles. In the document titled *Kennedy, John F.*, mention of *Kennedy* refers to the title, whereas other members of the family are named explicitly (e.g., *Joseph P.Kennedy*) so as not to confuse the reader.

After forming answer hypotheses, MURAX links together any equivalent answer hypotheses, and then rescores all hypotheses. The most general answer hypothesis in an equivalence class (usually the one that serves as a title) is assigned the cumulative score of all the members and used as the representative of the set. When a document title can be used as the representative, it usually provides the best description for the user. For example, for sentence (1) in section 6.1.1, "Harold" is a correct hypothesis, which is better represented by the more definitive title "Harold II."

Like surnames, the pronouns *he* and *she* often refer to the article's title (if the title is a person's name) or to an immediately preceding name. For the example output shown in Table 8, in the document displayed for Norman Mailer, the same answer hypothesis *Mailer* is used in both sentences; however, if he had been referred to instead as *he*, the result would be the same.

To verify whether an answer hypothesis refers to a person's name, MURAX first examines primary documents. If this attempt at verification fails, MURAX runs a secondary query If a title in the resulting matches is equivalent to the hypothesis and the title can be verified as a person's name, then the hypothesis is linked to the title.

6.4 Combining Phrase Matches

MURAX uses the various phrase matching procedures in concert to provide a ranked list of answer hypotheses. It does this by providing several intermediate layers of scoring and preference criteria. A list of criteria for partially ordering hypotheses is given below, in order of precedence. Typically, not all of these criteria apply for a given question; however, the lowest level criterion (criterion 5) always applies, and usually several others do as well.

1. For an input question that includes a type phrase, the highest ranking answer hypotheses are those with minimum mismatch to the type phrase.
2. Answer hypotheses are ranked according to the number of question phrases with which they co-occur. This is qualified by the number of different articles needed to match the most question phrases.
3. Predicate/argument matches are used to produce preferences among different answer hypotheses.
4. For *who* questions, an answer hypothesis that is verified as a person takes precedence.
5. Answer hypotheses are ranked in terms of their cooccurrence with question phrases.

7. Performance Evaluation

This section presents an evaluation of MURAX's performance for questions beginning with "who" and "what" that have simple noun phrase answers. (Questions having conjoined noun phrases such as "Richard Nixon and Nikita Khrushchev" are not included.)

Seventy questions taken from the game "Trivial Pursuit," each of which was known to have a simple noun phrase answer, were used for the evaluation. Additionally, it was confirmed that the answers to these questions were present in the encyclopedia--that is, given a question and its answer, a person could find text in the encyclopedia from which the answer could be inferred using common sense.

The evaluation was based on an objective, system-dependent criterion: the rank of the correct answer hypothesis. The results are shown in Table 10 below.

Table 10:

MURAX Evaluation Results

Rank of Correct Hypothesis	Number of Questions
Top	37
Top 3	49
Top 5	52
Not in Top 5	18

Table 10 indicates that MURAX's best guess is the correct answer for 53 percent of the questions (37 out of 70) and that the correct answer lies in MURAX's top five guesses for 74 percent of the questions. Using a cutoff of five guesses, answers are thus considered to be found for 74 percent of the questions and the mean rank of the correct hypothesis is 1.44.

The examples given in Table 5 above are typical of the questions that were used for the evaluation. Correct answers were in the top five hypotheses for all these questions of except question 10. For question 10, the answer "forget-me-not" was not found because MURAX does not consider references involving impersonal pronouns. Such a reference is necessary to infer the answer from the sentence "It is the state flower of Alaska. "

PART IV. CONCLUSION

The present invention provides a method for information retrieval based on the content of documents. In one embodiment of the invention, a natural-language input string and primary documents are accepted and analyzed by an answer extraction subsystem. Preliminary hypotheses are generated based on text contained in the primary documents. These hypotheses need not appear anywhere in the input string. Hypotheses are verified based on linguistic information from the input string, text from the primary documents, and text from secondary documents retrieved from a text corpus in response to secondary queries based on the hypotheses. Hypotheses are linked, scored, and ranked, and a best hypothesis is chosen. Scoring and ranking of hypotheses is based in part on verification evidence obtained for the hypotheses and can integrate information obtained from multiple documents. The method can be used either by itself or in the context of other, larger methods. In particular it can be used in conjunction with answer extraction in a two-phase IR method such as that embodied in MURAX.

Although the above is a complete description of certain embodiments and aspects of the invention, various alternatives, modifications, and equivalents can be used. For example, in the context of the MURAX system--which itself is but one of many possible embodiments of the method of the invention--enhancements can be made to improve performance within the scope of the present invention One such addition is the incorporation of synonym and hyponym information. In particular the WordNet thesaurus (G.A. Miller, R. Beckwith, C. Fellbaum, D. Gross, and K. Miller, "Five papers on WordNet," technical report, Princeton University Computer Science Laboratory, July 1990) appears well-suited to the MURAX task and could provide the system with useful synonym and hyponym information. For example, consider the question "What Pulitzer Prize-winning novelist ran

for mayor of New York City?" WordNet indicates that *novelist* is a hyponym of *writer*, *author* and also *person*. This means that the answer to the question is likely to be a person's name even though the question starts with "what". Furthermore, any type-phrase matches involving the words *writer* or *author* are also relevant.

Another contemplated addition to MURAX, again within the scope of the invention, involves using alternate grammar formalisms for linguistic analysis. The linguistic analysis in the version of MURAX described above is based on an underlying regular grammar formalism, both in the HMM tagger and the phrase recognizers. There can be benefits from the use of stochastic context-free grammars, which can also be trained from unlabelled text (see J M Kupiec, "Hidden Markov estimation for unrestricted stochastic context-free grammars," in Proceedings of the 1992 International Conference on Acoustics, Speech and Signal Processing, pages I-177-180, IEEE Signal Processing Society, IEEE, March 1992) and enable ambiguity to be quantified in probabilistic terms.

These additions to MURAX in no way exhaust the possibilities for extending MURAX within the scope of the present invention. For example, MURAX could be extended to include a comprehensive evaluation phase that takes account of not only the rank of correct hypotheses, but also the suitability of the match sentences that are presented to the user to verify phrase relations. Inasmuch as numerous extensions to MURAX are possible, and inasmuch as MURAX and its extensions represent only a small subset of the possible embodiments of the invention, the above description should not be taken as limiting the invention's scope. Rather, the scope of the invention is defined by the appended claims along with the full scope of equivalents to which such claims are entitled.

Claims

1. A method for organizing information retrieval based on the content of a set of documents, the method including using a processor to:
   accept the set of documents;
   analyze the content of at least one document of the set; and
   generate at least one answer hypothesis based on document content thus analyzed.

2. A method as claimed in claim 1, including using a processor to accept an input string in which the answer hypothesis is not present.

3. A method as claimed in claim 1 or claim 2, wherein a processor is used to verify the answer hypothesis, the processor gathering evidence for the answer hypotheses and additionally using a processor to determine a best answer hypothesis based on the evidence thus gathered.

4. A method as claimed in claim 3, in which a text corpus (12) is used to provide verification evidence, the text corpus (12) comprising the documents of the set of documents, or comprising documents not in the set of documents, or comprising the documents of the set and additional documents not in the set.

5. A method for retrieving documents from the text corpus (12) in response to a user-supplied natural language input string comprising words and a set of primary documents, the method including
   using a user interface (7) to accept the input string into an answer extraction subsystem (15);
   using a processor (5) to analyze the input string to detect phrases therein;
   using the processor (5) to accept the primary documents into the answer extraction subsystem (15); and
   using the answer extraction subsystem (15) to analyze the primary documents to detect additional phrases therein, the additional phrases not being present in the input string.

6. A method as claimed in claim 5, including using the answer extraction subsystem to verify the additional phrases as answer hypotheses.

7. A method of responding to an input string input into a system for computerized information retrieval, the method including using the system or a subsystem thereof to:
   accept the input string;
   accept a set of primary documents;
   detect phrases in the primary documents;
   generate preliminary hypotheses based on phrases so detected;
   select preliminary hypotheses as answer hypotheses for verification;

determine linguistic relations implied by the input string;
gather verification evidence for answer hypotheses; and
rank answer hypotheses

8. A method as claimed in claim 7, further including using the system or a subsystem thereof to link sets of equivalent hypotheses.

9. A method as claimed in claim 7 or claim 8, including using the system or a subsystem thereof to rank hypotheses by assigning scores to hypotheses; and
ordering hypotheses according to the scores thus assigned.

10. A method as claimed in any one of claims 7 to 8, in which the step of using the system or a subsystem thereof to rank hypotheses comprises using the system or a subsystem thereof to take into account at least some of the verification evidence; and/or to take into account co-occurrence of hypotheses with input string phrases in primary documents; and/or to take into account preliminary scores for preliminary hypotheses.

11. A computerized information retrieval method comprising the steps of:
using a processor to accept an input string and a set of primary documents;
using the processor to determine phrases in the input string;
using the processor to generate hypotheses not present in the input string based on text in the primary documents;
using the processor to verify the hypotheses using the primary documents; and
using the processor to score the hypotheses.

12. A method for organizing information retrieval based on the content of a plurality of documents accepted by a processor

13. A method for using a processor in a system to organize information retrieval based on the content of documents of a text corpus, the method for organizing computerized information retrieval being based on an answer hypothesis generated by the system.

14. A twophase method for use in a system for processing an input string supplied by a user, the method including
using the system to construct and execute a series of primary queries based on shallow linguistic analysis of the input string in order to retrieve primary documents; and
using the system to perform answer extraction, the answer extraction comprising hypothesis generation and secondary query construction and execution.

15. A method for retrieving documents from the text corpus in response to a usersupplied natural language input string comprising words, the method including
using a user interface (7) to accept the input string into a primary query construction subsystem (10);
using the primary query construction subsystem (10) to analyze the input string to detect phrases therein;
using the primary query construction subsystem (10) to construct a series of queries based on the detected phrases, the queries of the series being constructed automatically by the primary query construction subsystem (10) through a sequence of operations that comprises successive broadening and narrowing operations;
using the primary query construction subsystem (10), the information retrieval subsystem, a text corpus (12), and channels to execute the queries of the series;
using the primary query construction subsystem to rank documents retrieved from the text corpus in response to one or more queries thus executed to produce a set of primary documents;
using a channel to send the primary documents, the input string, and the phrases detected in the input string from the primary query construction subsystem to the answer extraction subsystem;
using the answer extraction subsystem to generate hypotheses not present in the input string based on text in the primary documents;
using the answer extraction subsystem to verify the hypotheses using the primary documents; and
using the answer extraction subsystem to score the hypotheses.

16. A system for information retrieval, the system comprising a processor, characterised in that the system utilises a method as claimed in any one of claims 1 to 15.

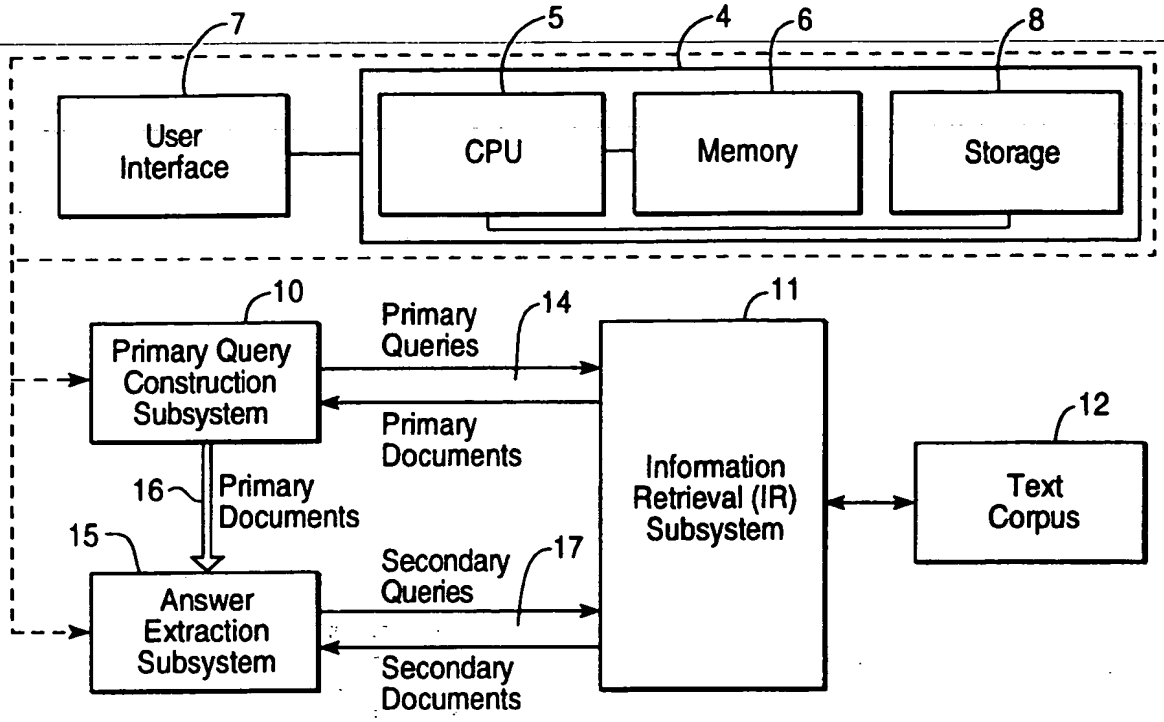
7
5
4
6
8
User Interface
CPU
Memory
Storage
10
Primary Query Construction Subsystem
Primary Queries
14
Primary Documents
11
Information Retrieval (IR) Subsystem
12
Text Corpus
16
Primary Documents
15
Answer Extraction Subsystem
Secondary Queries
17
Secondary Documents

FIG. 1

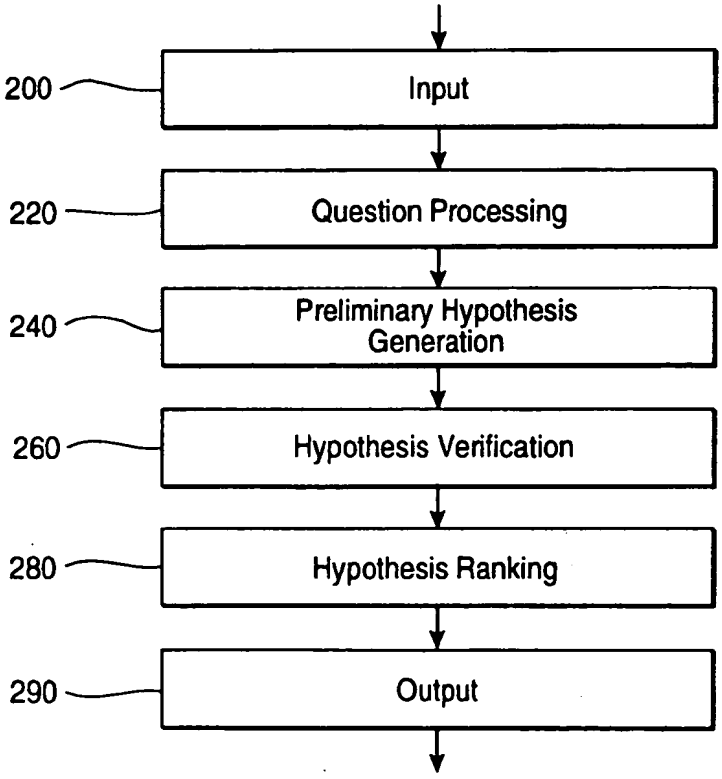
200
Input
220
Question Processing
240
Preliminary Hypothesis Generation
260
Hypothesis Verification
280
Hypothesis Ranking
290
Output

FIG. 2

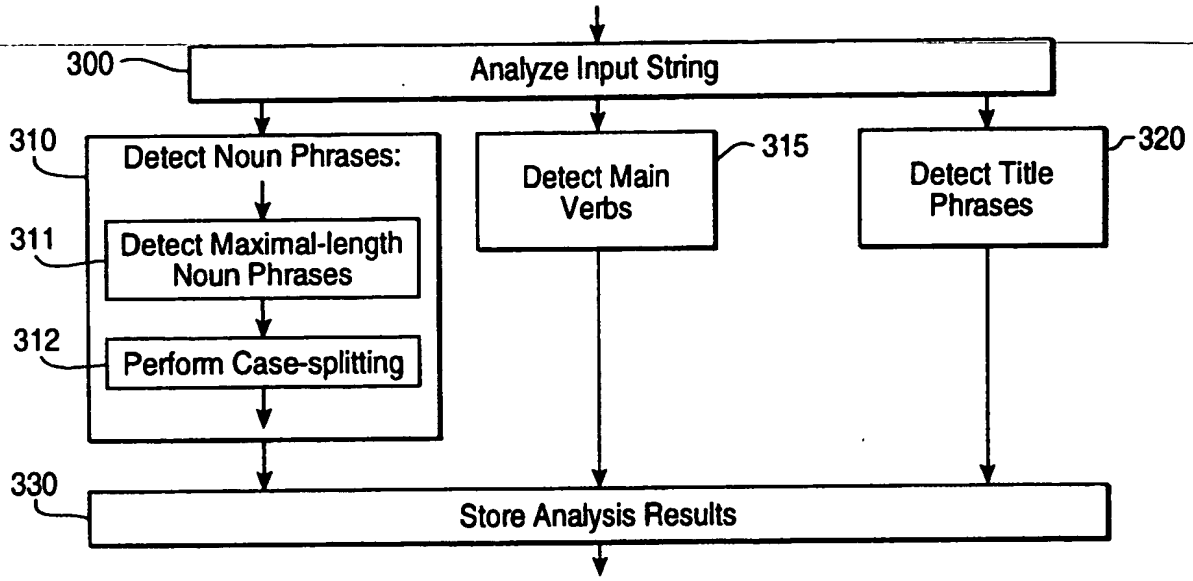

FIG. 3

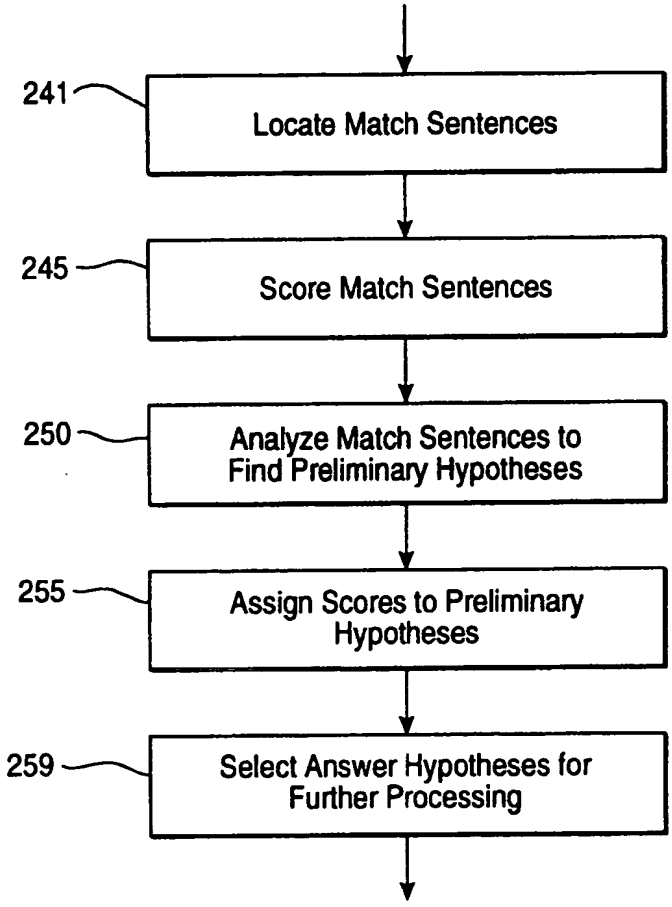

FIG. 4

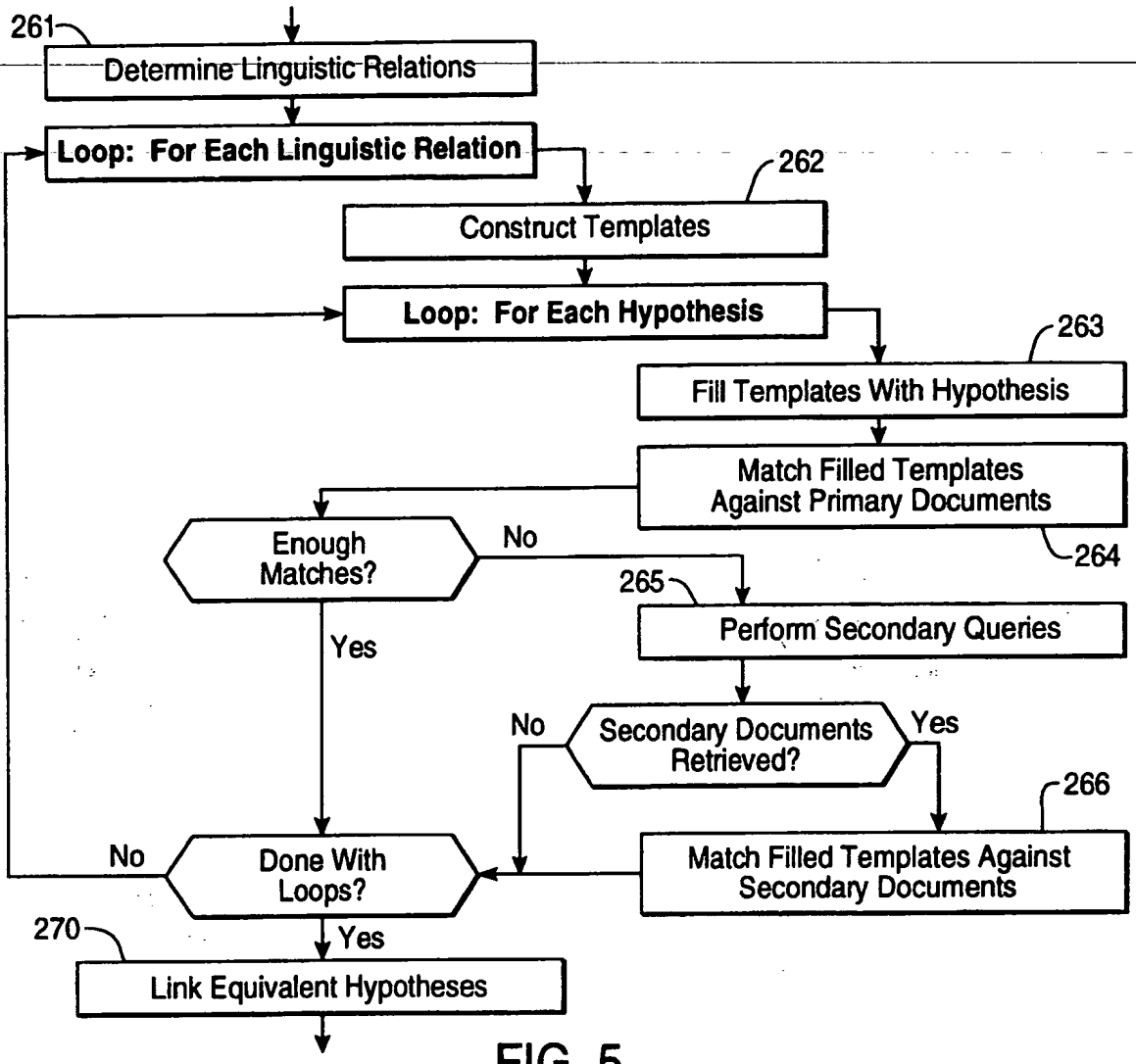
261
Determine Linguistic Relations
Loop: For Each Linguistic Relation
262
Construct Templates
Loop: For Each Hypothesis
263
Fill Templates With Hypothesis
Match Filled Templates Against Primary Documents
264
Enough Matches?
No
265
Perform Secondary Queries
Yes
No
Secondary Documents Retrieved?
Yes
266
No
Done With Loops?
Match Filled Templates Against Secondary Documents
270
Yes
Link Equivalent Hypotheses

FIG. 5

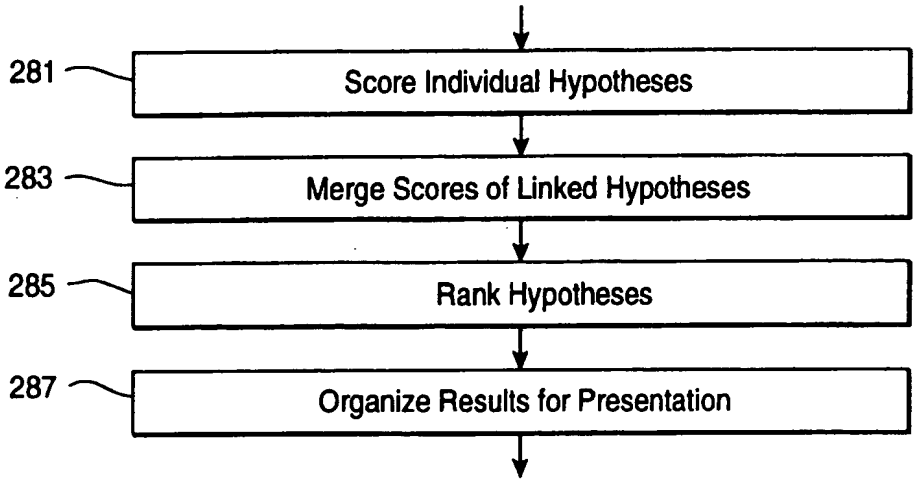
281
Score Individual Hypotheses
283
Merge Scores of Linked Hypotheses
285
Rank Hypotheses
287
Organize Results for Presentation

FIG. 6